Equipping Retrieval-Augmented Large Language Models with Document Structure Awareness

Lingnan Xu[†,‡], **Chong Feng**[†,§,*], **Kaiyuan Zhang**[†],
Liu Zhengyong[‡], **Wenqiang Xu**[‡], **Fanqing Meng**[†]

[†]School of Computer Science, Beijing Institute of Technology [‡]Ant Group
[§]Southeast Academy of Information Technology, Beijing Institute of Technology

{xln, fengchong, zky, mengfanqing}@bit.edu.cn,

{xulingnan.xln, liuzhengyong.lzy, yugong.xwq}@antgroup.com

Abstract

While large language models (LLMs) demonstrate impressive capabilities, their reliance on parametric knowledge often leads to factual inaccuracies. Retrieval-Augmented Generation (RAG) mitigates this by leveraging external documents, yet existing approaches treat retrieved passages as isolated chunks, ignoring valuable structure that is crucial for document organization. Motivated by this gap, we propose *Retrieve-DocumentRoute-Read* (**RDR**^{2}), a novel framework that explicitly incorporates structural information throughout the RAG process. RDR^2 employs an LLM-based router to dynamically navigate document structure trees, jointly evaluating content relevance and hierarchical relationships to assemble optimal evidence. Our key innovation lies in formulating document routing as a trainable task, with automatic action curation and structure-aware passage selection inspired by human reading strategies. Through comprehensive evaluation on five challenging datasets, RDR^2 achieves state-of-the-art performance, demonstrating that explicit structural awareness significantly enhances RAG systems' ability to acquire and utilize knowledge, particularly in complex scenarios requiring multi-document synthesis.[1]

1 Introduction

Large language models (LLMs) (Brown et al., 2020) have demonstrated remarkable capabilities across a wide range of natural language processing (NLP) tasks, yet even state-of-the-art models continue to generate factually incorrect responses (Mallen et al., 2023; Min et al., 2023; Ji et al., 2023) despite their growing scale and capability (Ouyang et al., 2022). Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Guu et al., 2020; Borgeaud et al., 2022) addresses these limitations through a *Retrieve-and-Read* paradigm,

which first retrieves relevant passages then uses them as context for generation (Lewis et al., 2020; Izacard and Grave, 2021; Jiang et al., 2022; Shi et al., 2024). This approach combines the strengths of information retrieval and generative models, proving particularly effective for atomic-fact question answering (QA) (Joshi et al., 2017; Thorne et al., 2018; Kwiatkowski et al., 2019; Mallen et al., 2023) where *a single precise retrieval suffices to answer clear information needs*.

Recent advances in RAG have extended its capabilities to complex knowledge-intensive scenarios requiring multi-perspective responses, particularly for factual-inductive queries that demand coherent synthesis of multiple knowledge fragments (Fan et al., 2019; Stelmakh et al., 2022; Amouyal et al., 2023). However, current RAG frameworks process retrieved passages as isolated chunks, discarding their inherent document structure - a limitation stemming from both *structure-agnostic pipeline design* and the *flat-context paradigm of standard retrieval methods*.

While fixed chunking ensures retrieval efficiency, it restricts query-adaptive content selection, discarding the document's native organization which humans naturally exploit for information navigation and relational reasoning. At the reading phase, retrieved passages are simply ordered by relevance scores, potentially disrupting their original sequence in the source document. Even with useful information, this loss of structural priors forces the model to implicitly reconstruct relationships that were explicitly encoded in the source hierarchy. This structural blindness constrains RAG's knowledge acquisition and synthesis capabilities.

In this paper we ask: *can LLMs leverage document structural information*, and *can RAG systems benefit from such structural awareness*? We propose *Retrieve-DocumentRoute-Read* (**RDR**^{2}), where a structure-aware LLM performs document

---

24608

routing through three actions inspired by how humans selectively read sections, expand promising headings, and skip irrelevant parts when browsing articles. Through this process, RDR^2 dynamically assembles query-oriented passages for better knowledge acquisition and utilization.

We evaluate RDR^2 on five QA datasets representing diverse formats: TriviaQA (Joshi et al., 2017) (single-answer), HotpotQA (Yang et al., 2018) (multi-hop), QAMPARI (Amouyal et al., 2023) (list-style), ASQA (Stelmakh et al., 2022) (ambiguous), and ELI5 (Fan et al., 2019) (in-depth). As shown in Figure 1, RDR^2 achieves new state-of-the-art results with only the router trained on questions from the ASQA training set (without answer supervision), while keeping the retriever and reader off-the-shelf. Additionally, RDR^2 enables test-time scaling without weight updates and demonstrates generalization across different RAG components (i.e., retrievers and readers).

Our main contributions are:

- The proposal of RDR^2, the first RAG framework explicitly incorporates document structure throughout the retrieval and reading process, to enhance both knowledge acquisition and utilization;
- A novel formulation of document routing as a trainable task, with an automatic action curation pipeline and LLM-based router training;
- Comprehensive experiments on five datasets establishing RDR^2's consistent superiority over state-of-the-art methods.
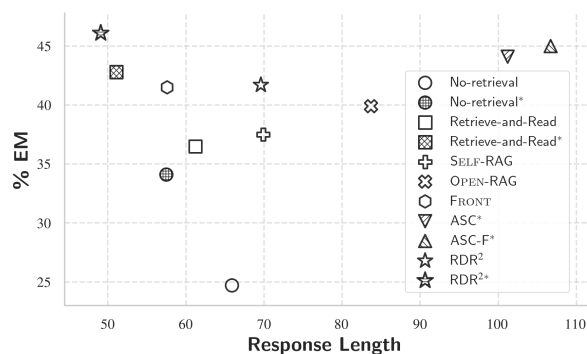

Figure 1: Performance comparison on ASQA, where RDR^2 achieves the highest Exact Match (EM) score while generating the most concise responses. Readers are based on either Llama-2-13B or ChatGPT (*).

2 Related Work

Retrieval-Augmented Generation (Lewis et al., 2020; Guu et al., 2020; Borgeaud et al., 2022)

(RAG) augments language models with non-parametric knowledge through retrieved passages, demonstrating significant improvements in knowledge-intensive tasks (Ram et al., 2023; Asai et al., 2023a). The standard *Retrieve-and-Read* framework operates in two stages: (1) a dense retriever, typically employing a bi-encoder architecture (Karpukhin et al., 2020; Ni et al., 2022; Wang et al., 2024), retrieves passages relevant to the input question, and (2) an LM reader processes these passages, either as an off-the-shelf model (Zhou et al., 2024; Li et al., 2025) or through task-specific fine-tuning (Izacard et al., 2023; Lin et al., 2023; Jain et al., 2023; LUO et al., 2024; Gan et al., 2024), to generate grounded responses. While effective for simple tasks with clear information needs, RAG systems show limitations in complex scenarios, necessitating more advanced methods.

Knowledge Acquisition. To achieve more comprehensive knowledge acquisition, recent works develop enhanced retrieval mechanisms. FLARE (Jiang et al., 2023) prompts an LLM to actively decide when and what to retrieve based on the model's confidence (i.e., token probabilities). Ma et al. (2023) introduces query rewriting to bridge the gap between user questions and retrieval requirements. CoRAG (Wang et al., 2025) fine-tunes an LLM to generate intermediate retrieval chains, enabling step-by-step multi-hop querying. Unlike prior works that focus on pre-retrieval query optimization, our approach enhances knowledge acquisition through post-retrieval document routing - iteratively exploring document hierarchies to uncover useful information.

Knowledge Utilization. For knowledge utilization, effective RAG requires critical evaluation and integration of retrieved knowledge. SELF-RAG (Asai et al., 2023b) fine-tunes LLMs to critique retrieved passages via self-reflection, assessing their relevance, supportiveness, and utility. RankRAG (Yu et al., 2024) instruction-tunes a single LLM for the dual purpose of context ranking and answer generation, improving end-to-end knowledge grounding. Departing from static chunk filtering, our method dynamically assembles node-level information units within document hierarchy, achieving both structural integrity and adaptive flexibility.

Structural Information. Several approaches have attempted to incorporate structural information into RAG frameworks. GraphRAG (Edge et al., 2024) processes documents into a knowl-
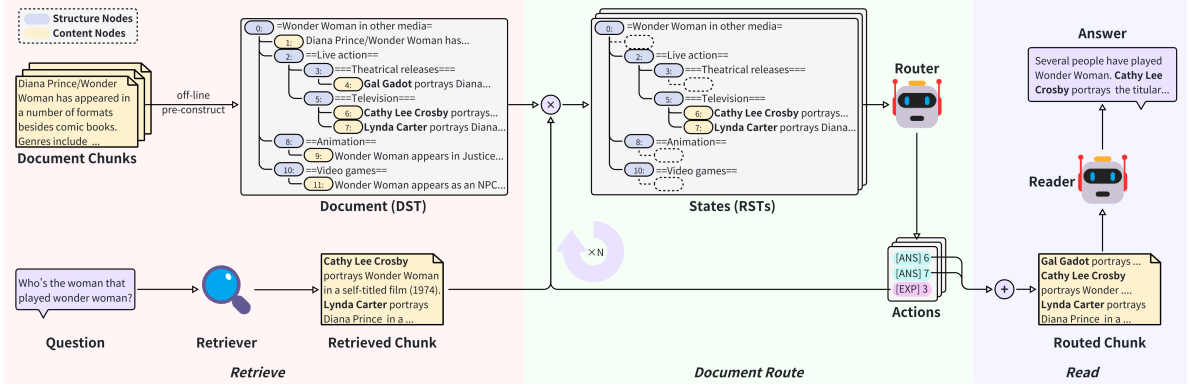
Figure 2: Overview of the RDR^2 framework. RDR^2 extends standard *Retrieve-and-Read* with document-structure-aware routing for iterative, fine-grained knowledge retrieval. *Retrieve*: input question q, output retrieved chunks C_{re}; *Document Route*: input q, C_{re} and corresponding documents D, output routed chunks C_{ro}; *Read*: input q and C_{ro}, output final answer a.

edge graph with hierarchical community summaries, establishing a RAG paradigm distinct from semantic retrieval over flat text chunk. RAPTOR (Sarthi et al., 2024) constructs hierarchical document embeddings through recursive node-level clustering and summarization, capturing progressively abstracted semantic content across tree levels. While existing approaches *offline-encode* hierarchical information into *fixed representations* (e.g., summaries or embeddings), our framework *online-perceives* document structure through *dynamic routing*.

3 Methodology

In this section, we present RDR^2 (*Retrieve-DocumentRoute-Read*), a novel framework that endows the retrieval-augmented systems with explicit awareness of document structure. We first introduce the overview of our framework. Then we define tree structures to represent the document hierarchy, ensuring stable scope and adaptive contextual focus. Finally, we introduce the document routing task and the scheme for training a structure-aware LLM router.

3.1 *Retrieve-DocumentRoute-Read*

As illustrated in Figure 2, the RDR^2 framework consists of three stages:

Retrieve. Given an input question q and a datastore $\mathcal{D}$, the Retriever retrieves the top-k most relevant chunks $C_{re} = \{c_{re}^{(1)}, \cdots, c_{re}^{(k)}\}$, along with their originating documents $D = \{d_1, \cdots, d_k\}$.

$$\{\langle c_{re}^{(i)}, d_i \rangle\}_{i=1}^{k} = \text{Retriever}(q, \mathcal{D}) \qquad (1)$$

Document Route. This stage transforms chunk-wise retrieved results into document-wise routed chunks $C_{ro} = \{c_{ro}^{(1)}, \cdots, c_{ro}^{(m)}\}$ through an iterative process, where an LLM-based Router selectively expand relevant sections while maintaining awareness of the document's organizational framework. At each step t, the Router takes the question q and the current routing state $s_i^{(t)}$ to decide a series of actions, where each element consists of a ternary action tag $a_{ij}^{(t)} \in \{[\text{ANS}], [\text{EXP}], [\text{REF}]\}$[2], along with a selected passage node $p_{ij}^{(t)}$.

$$\{\langle a_{ij}^{(t)}, p_{ij}^{(t)} \rangle\}_{j=1}^{n_i^{(t)}} = \text{Router}(q, s_i^{(t)}) \qquad (2)$$

The routing state (i.e. Retrieval SubTree in Section 3.2) encapsulates structural information, enabling the Router to navigate through the document hierarchy. Each document d_i's routing state $s_i^{(t)}$ is initialized with the corresponding retrieved chunks $C_{re}^{(i)} \subseteq C_{re}$ and updated by the Router actions of the previous round $\{\langle a_{ij}^{(t-1)}, p_{ij}^{(t-1)} \rangle\}_{j=1}^{n_i^{(t-1)}}$. Here, the operator $\otimes$ denotes content expansion:

$$s_i^{(t)} = d_i \otimes \begin{cases} E_i^{(t-1)} & , t > 1 \\ C_{re}^{(i)} & , t = 1 \end{cases} \qquad (3)$$

$$E_i^{(t)} = \{p_{ij}^{(t)} \mid j \in \{1, ..., n_i^{(t)}\}, a_{ij}^{(t)} = [\text{EXP}]\} \qquad (4)$$

The routed chunk $c_{ro}^{(i)} \in C_{ro}$ is accumulated by aggregating the selected passages across all routing

---

[2][ANS]: extracting useful contents to *answer*; [EXP]: unfolding promising titles to *expand*; [REF]: stopping the routing process to *refuse*.
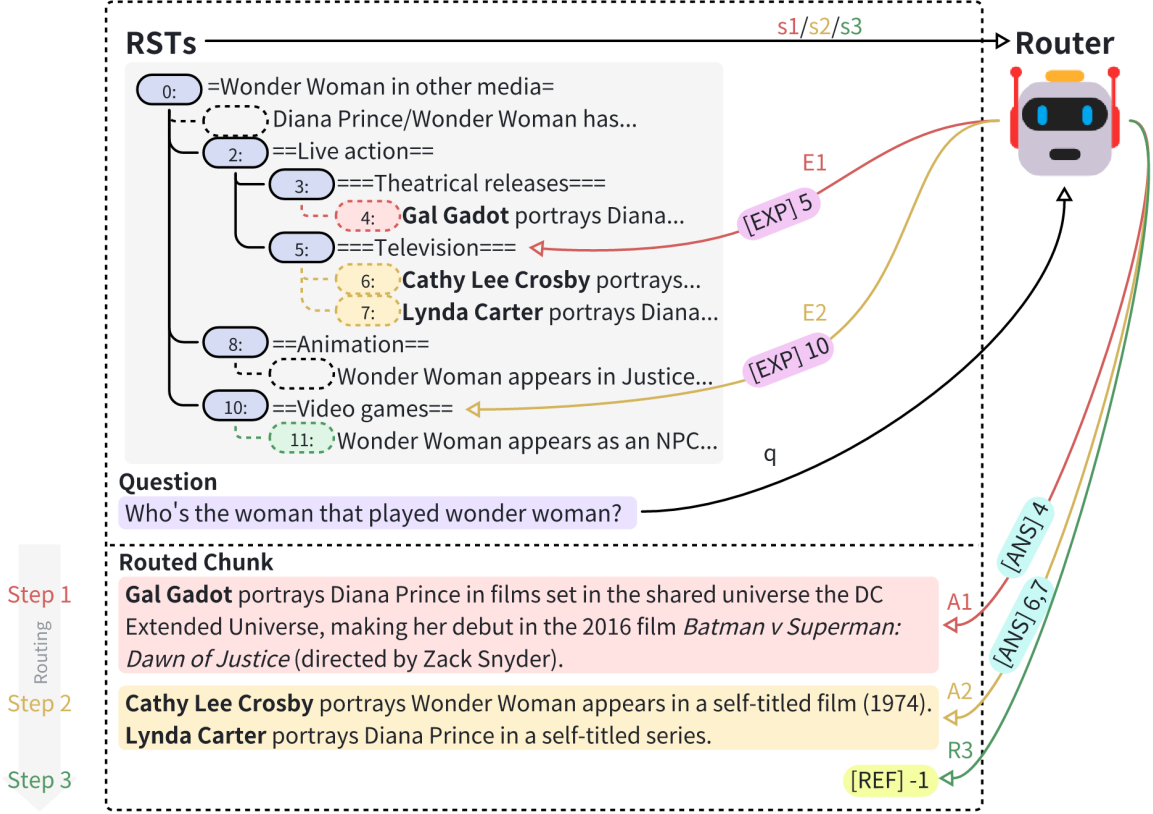
Figure 3: Workflow of the routing module. Given a user input q and a document structure tree (Section 3.2) anchored by retrieved chunks, RDR^2 maintains a retrieval subtree s where: (i) all structure nodes persist, (ii) only content nodes under currently selected headings are expanded (previous fold). At step t, the router generates action $\{\langle a_j^{(t)}, p_j^{(t)} \rangle\}_{j=1}^{n_t} = \text{Router}(q, s_t)$ to: (a) select useful content nodes, (b) unfold a promising structure node, or (c) stops routing.

steps $t = \{1, \cdots, T_i\}$.

$$c_{ro}^{(i)} = \bigoplus_{t=1}^{T_i} A_i^{(t)} \qquad (5)$$

$$A_i^{(t)} = \{p_{ij}^{(t)} \mid j \in \{1, ..., n_i^{(t)}\}, \ a_{ij}^{(t)} = [\text{ANS}]\} \qquad (6)$$

where the operator $\oplus$ denotes passage concatenation.

Read. The Reader (typically an LLM) generates the final answer a, conditioning on both the input question q and the routed passages C_{ro}.

$$a = \text{Reader}(q, [c_{ro}^{(1)}, \cdots, c_{ro}^{(m)}]) \qquad (7)$$

3.2 Document Structure Representation

While standard RAG frameworks process only flat content chunks, our approach preserves critical structural information through formal tree representations. To capture hierarchical relationships in documents, we define two types of nodes:

(1) *Structure Nodes* represent organizational hierarchy (e.g., headings), and (2) *Content Nodes* contain substantive textual information (e.g., passages).

Document Structure Tree. A Document Structure Tree (DST) encodes the full document hierarchy, where each node is represented as:

$$\text{DST-node} = \langle id, text, \tau, parent, \mathcal{C} \rangle \qquad (8)$$

Here $\tau \in \{\text{structure}, \text{content}\}$ denotes the node type, and $\mathcal{C}$ indicates the ordered set of child nodes. Each node is defined by a unique identifier (*id*), associated *text* content - either a heading title (for structure nodes) or passage text (for content nodes) - and a pointer to its *parent* node (null for the root). The root node, always a structure node, corresponds to the document title.

Retrieval Subtree. A Retrieval SubTree (RST) is derived from the DST, designed to maintain stable retrieval scope while adaptively updating contextual focus. An RST consists of (1) all structure

nodes (complete document hierarchy), and (2) selected content nodes (partial content coverage).

During inference, the RST is first initialized with the retrieved passages along with their content siblings, then iteratively updated by replacing them with previously unseen content nodes under a single router-selected heading, while preserving all structure nodes (see Algorithm 1 in Appendix A.1). This constrained derivation strategy ensures stable RST size while dynamically refining the contextual focus.

3.3 Routing Module

As shown in Figure 3, the routing module synergistically combines document tree structure with an LLM-based router, enabling structure-aware retrieval-augmented generation.

Task Formulation. We define document routing task as iterative navigation through a document structure tree, dynamically assembling fine-grained passage chunks with both content relevance and structural integrity. This process emerges through compositional application of three atomic actions at each step:

- [ANS]: Select a visible *content node* when its text directly answers the question;
- [EXP]: Unfold a collapsed *structure node* if its heading text or contextual position suggests potential relevance;
- [REF]: Stop exploring the current subtree when no nodes satisfy [ANS] or [EXP] criteria.

Action Curation. Standard RAG datasets consists of a question with a reference answer, without providing the intermediate routing trajectories. We propose an automatic method for curating routing actions *solely* from the question, requiring no necessary access to the answer. Specifically, given a question q, we first retrieve top-k passages via an off-the-shelf retriever, access their originating documents, and derive corresponding retrieval subtrees S. We condition an LLM respectively on each subtree $s \in S$, along with the question q to generate single-turn routing actions A. Finally, the routing dataset cruated consists of $\langle q, s, A \rangle$ triples.

Training. The training paradigm focuses on equipping the model with fundamental decision-making capabilities through exposure to individual routing actions (as opposed to complete iterative procedures). We fine-tune an LLM on the curated routing dataset using the standard next-token-prediction objective under supervised-fine-tuning

(SFT), where the cross-entropy loss $\mathcal{L}$ is computed only on the target output tokens. This approach provides the necessary components for multi-step exploration during inference.

$$\mathcal{L} = - \log P(A|q, s) \tag{9}$$

We convert document hierarchy into LLM-understandable text representation. Specifically, the input retrieval subtree uses the newline-delimited `"id: text"` format, where each level of hierarchy is represented by an additional indentation unit preceding the node identifier. The output action follows the `"[ACTION] id: text_prefix"` format to ensure semantic grounding to the original id-text binding.

4 Experiments

4.1 Datasets and Metrics

We evaluate RDR^2 on five knowledge-intensive tasks across different QA formats. We follow previous works (Gao et al., 2023) to randomly sub-sample at most 1,000 examples from each dataset due to the experimental cost. Across all datasets, only the question field is used for both retrieval and generation, with Wikipedia consistently serving as the retrieval datastore.

Short-form Generation. **TriviaQA** (Joshi et al., 2017) consists of trivia questions, each calling for a single short answer. **HotpotQA** (Yang et al., 2018) features Wikipedia-based question-anwer pairs requiring interleave retrieval and reasoning. For both datasets, we report *EM* Recall and String F_1, following standard setups in Asai et al. (2023b) and Wang et al. (2025).

QAMPARI (Amouyal et al., 2023) is a list-style QA dataset where answers comprise multiple factual short entities (avg. 13 instances) originated from diverse passages. We report F_1-5, *precision* and *recall-5*, where recall-5 is considered 100% if at least five gold answers are covered, following ALCE (Gao et al., 2023) benchmark.

Long-form Generation. **ASQA** (Stelmakh et al., 2022) is a long-form factoid QA dataset featuring inherently ambiguous questions that requires RAG methods to reconcile diverse interpretations and produce coherent responses (avg. 65 words). We adopt the official metrics from the ASQA paper: *EM* (Exact Match), *ROUGE-L*, and *Disambig-F_1*.

ELI5 (Fan et al., 2019) contains complex, diverse, open-ended questions derived from post ti-
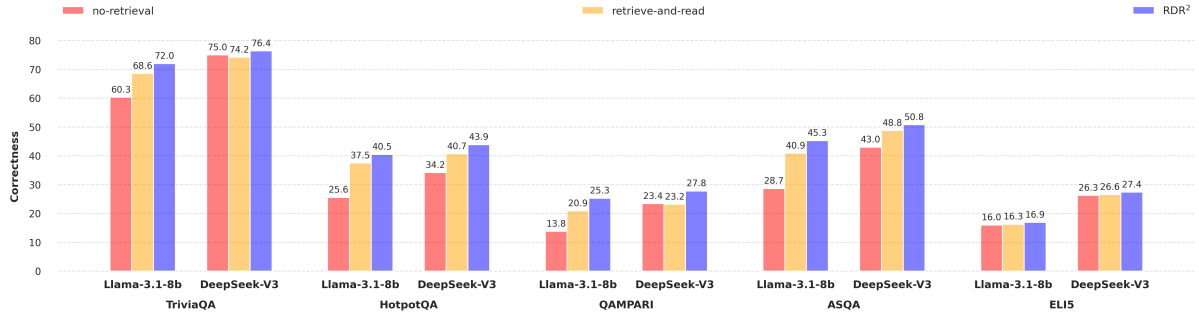
Figure 4: Comparison between RDR^2 and baselines across all datasets with different readers. We report the primary correctness metric for each dataset: Exact Match for TriviaQA, HotpotQA and ASQA, F_1-5 for QAMPARI and Claim Recall for ELI5.

tles in Reddit's "Explain Like I'm Five" forum, requiring systems to retrieve multiple documents and elaborate in-depth explanations (avg. 131 words). Following Gao et al. (2023), we use *Claim Recall*, computed by checking whether the generated output entails reference sub-claims using an NLI model.

We additionally evaluate fluency and conciseness for long-form generation tasks. For fluency, we follow ALCE to use *MAUVE* (Pillutla et al., 2021) to assess distributional similarity between generated and ground-truth answers. For conciseness, we report response *length* (in words), as longer outputs may artificially boost recall-type metrics (e.g., EM or Claim Recall).

4.2 Baselines

We evaluate our framework against three categories of baselines: (1) *No-Retrieval*: the reader directly answers questions using only its parametric knowledge, (2) *Retrieve-and-Read*: the standard RAG pipeline with top-k retrieved passages, and (3) *Advanced RAG*: including methods based on proprietary LLMs: ASC and its variant ASC-F (Thirukovalluru et al., 2024), as well as techniques fine-tuned on open-source LLMs: SELF-REASONING (Xia et al., 2025), SELF-RAG[3] (Asai et al., 2023b), OPEN-RAG[4] (Islam et al., 2024), and FRONT (Huang et al., 2024).

4.3 Experimental Settings

For retrieval, we use the Wikipedia dump from Karpukhin et al. (2020). We construct DSTs (defined in Section 3.2) from the corresponding wiki

pages, totaling 5.82M documents. Unless otherwise specified, we use the off-the-shelf Contriever-MS MARCO (Izacard et al., 2022) as the retriever, with top-5 retrieved chunks for all retrieval-augmented methods.

We curate routing actions using Deepseek-v3 (Liu et al., 2024) following the procedure defined in Section 3.3 on ASQA training questions, resulting in 23,827 training samples and 500 test samples . The router is fine-tuned via LoRA (Hu et al., 2022) on Llama-3.1-8B-Instruct (Grattafiori et al., 2024) for 3.5 epochs, using LlamaFactory (Zheng et al., 2024) (see Appendix A.1 for data curation details, Appendix A.2 for training hyperparameters, and Appendix C for router prompts).

Llama-2-13B-Chat (Touvron et al., 2023) and Llama-3.1-8B-Instruct (Grattafiori et al., 2024) are used as the open-source readers. To ensure fair comparison, we apply greedy decoding with model-specific maximum tokens, as significant inter-model length variations were observed (consistent with Asai et al. (2023b)'s findings). For proprietary models including ChatGPT (Ouyang et al., 2022) and Deepseek-v3 (Liu et al., 2024), we set temperature=0.2 without length constraints, since their output lengths naturally align with the reference (see Appendix C for reader prompts).

All experiments run on single NVIDIA A100-PCIE-40GB GPUs.

5 Results and Analysis

We first present overall results, then perform ablation studies to assess the contribution of each key component. Finally, we examine the framework's scalability under different test-time conditions and its robustness to various RAG component choices. A full case study is provided in Appendix D.

---

[3]We increased the [Retrieve] token probability by 0.2 to promote multi-turn retrieval for a fair comparison.

[4]With only the 7B model publicly released, we fine-tuned the 13B variant using the official training script.

	QAMPARI			ASQA					ELI5		
	F_1-5	R-5	Pre	EM	D-F_1	R-L	Mau	Len	Cla	Mau	Len
Reader based on ChatGPT											
ASC-F	18.8	**45.0**	13.4	45.0	31.9	-	41.3	106.7	22.2	**22.7**	172.7
ASC	26.2	33.0	23.0	44.1	32.2	-	47.0	101.2	21.4	21.3	163.6
RDR^2(Ours)	**26.4**	29.0	**30.9**	**46.1**	**37.1**	**38.5**	**70.6**	49.1	**23.3**	14.4	155.2
Reader fine-tuned on Llama-2-13B											
Self-Reasoning	-	-	-	35.2	-	-	-	-	-	-	-
Self-RAG*	6.5	9.0	6.4	37.5	27.5	39.2	**77.7**	69.9	11.8	**42.8**	81.9
Open-RAG*	2.5	3.9	2.3	39.9	24.0	**40.4**	17.2	83.7	11.9	19.1	129.2
Front	-	11.9	22.6	41.5	-	38.6	76.1	57.6	9.3	34.4	75.1
RDR^2(Ours)	**23.2**	**24.3**	**25.0**	**41.7**	**31.6**	39.2	61.2	69.6	**15.4**	23.9	148.3

Table 1: Comparison between RDR^2 and other RAG methods on QAMPARI, ASQA and ELI5 wrt. corresponding metrics. F_1-5 is the harmonic mean of recall-5 (R-5) and precision (Pre), EM is Exact Match, D-F_1 is Disambig-F_1, R-L is ROUGE-L, Mau is MAUVE, Cla is Claim Recall. **Bold** indicates best results within each reader category. Gray denotes the word-level length (Len). * marks the results from our reproduction.

5.1 Main Results

Overall Performance. Figure 4 evaluates the overall performance of RDR^2 against two fundamental frameworks: *no-retrieval* and *Retrieve-and-Read*. Notably, in RDR^2 only the router is finetuned on ASQA training questions (without answer supervision), while both retriever and reader remain off-the-shelf. TriviaQA, HotpotQA, QAMPARI and ELI5 serve as challenging generalization tests, being completely withheld from our router training.

RDR^2 continuously improves RAG performance. With larger language models, standard *Retrieve-and-Read* shows diminishing returns over *no-retrieval*, suggesting their stronger parametric knowledge reduces reliance on retrieved content. While RDR^2 also exhibits this scaling trend versus *no-retrieval*, its improvement over *Retrieve-and-Read* remains relatively stable across model scales, confirming the inherent value of document structure awareness in retrieval-augmented generation.

RDR^2 effectively generalizes to held-out datasets. While RDR^2 maintains strong performance on QAMPARI comparable to its ASQA results, we observe limited gains on ELI5. This aligns with prior findings (Krishna et al., 2021; Jiang et al., 2023) on the intrinsic challenges of open-ended long-form QA, where the expansive space of potentially valid answers poses fundamental difficulties for retrieval-augmented approaches and their evaluation.

Comparison with baselines. Table 1 compares RDR^2 against cutting-edge RAG methods employing either proprietary LLMs (ChatGPT) or fine-tuned open-source Llama-2-13B variants as their backbone readers.

RDR^2 achieves new state-of-the-art results. Across all three datasets - QAMPARI, ASQA and ELI5 - RDR^2 consistently outperforms existing approaches, demonstrating strong generalization across diverse QA scenarios. Specifically:

It is noteworthy that among the compared methods based on open-source models, all require reader fine-tuning on carefully annotated question-answer pairs (some including training set of the downstream tasks), whereas our approach achieves superior performance using only readily available questions for router training, paired with an entirely off-the-shelf reader.

Furthermore, methods employing proprietary LLMs generate significantly longer responses ($2\times$ the gold answer length on ASQA) to achieve high EM recall, while our approach attains better results with approximately 50% shorter outputs. On QAMPARI, this verbosity leads to precision degradation, whereas our method maintains balanced precision-recall performance. These observations collectively validate our framework's enhanced efficiency in information delivery.

5.2 Ablation Study

Table 2 presents comprehensive ablation studies analyzing three critical dimensions of our framework: *pipeline architecture* (Section 3.1), *router information* (Section 3.2), and *routing actions*

(Section 3.3). We evaluate both intermediate routed passages and final generated answers, measuring factual correctness through Exact Match (EM) and verbosity via word count (Len).

	Passage		Answer	
	EM	**Len**	**EM**	**Len**
RDR^2(Ours)	<u>57.3</u>	104.2	**45.3**	71.3
w/o router	51.7	100.0	40.9	69.2
w/o structure	49.8	67.5	41.3	71.0
w/o similarity	54.8	100.9	<u>43.9</u>	72.3
w/o content	54.2	93.9	43.7	70.0
w/o [EXP]	52.9	81.7	42.5	71.9
w/o [REF]	**61.2**	176.3	42.9	70.7

Table 2: Ablation Study on ASQA. Ablated variants (w/o = without) are defined in Section 5.2. We report Exact Match (EM) and word-level length (Len) for passages and answers. **Bold** and <u>Underline</u> denote best and second best results, respectively.

5.2.1 Pipeline Architecture

Ablating Router. Removing the routing module (**w/o router**) reduces the RAG pipeline to standard *Retrieve-and-Read* framework. Our full framework significantly improves factual recall (+5.6 EM) while maintaining comparable passage length (104.2 vs. 100.0), demonstrating enhanced informativeness without compromising conciseness. This improvement carries through to answer generation (+4.4 EM), demonstrating consistent gains across the entire RAG pipeline.

5.2.2 Router Information

The router processes two types of information: (1) *structure* from document headings, and (2) *similarity* from retrieved passages. We ablate each component:

Ablating Structure. The **w/o structure** variant discards document hierarchy and use only retrieved passages[5], where the router simply accepts or refuses individual passages. We observe significant drops in both passage retrieval (-7.5 EM) and answer generation (-4.0 EM) versus the full framework, confirming structural cues provide critical gains. Compared to *w/o router*, this ablation yields less informative passages (-1.9 EM) but better answers (+0.4 EM), showing structural awareness enables more effective knowledge organization despite occasional over-filtering.

---

[5]To ensure fair comparison, we reconstruct content at the node level to avoid information loss from chunk truncation.

Ablating Similarity. The **w/o similarity** variant initializes the RST with content nodes under a *random* heading instead of retrieved passage siblings. A stricter variant (**w/o content**) removes content nodes entirely, despite this configuration being completely unseen during training. Ablating similarity causes moderate performance drops (-2.5 EM passages, -1.4 EM answers), confirming that providing question-relevant content offers crucial guidance for structural understanding and document routing. The small gap between these variants (0.6 EM passages, 0.2 EM answers) demonstrates the router's trained structural reasoning generalizes to unseen document formats.

5.2.3 Routing Actions

We validate each atomic action's necessity for document routing:

Ablating [EXP]. The router can only select or refuse among currently visible nodes, losing the ability to explore new subtrees (**w/o [EXP]**). The noticeable declines versus full framework (-4.4 passage EM, -2.8 answer EM) confirms expansion is crucial for discovering content that can hardly be recalled by similarity alone. Yet still outperforms *w/o router* (+1.2 passage EM, +1.6 answer EM), showing RAG can benefit from basic structure awareness.

Ablating [REF]. The router must either answer or expand at least one node in each step, potentially forcing suboptimal choices (**w/o [REF]**). Passage informativeness is substantially increased (+3.9 EM), yet its length doubled, introducing noise that ultimately harms answer quality (-2.4 EM), proving selective rejection is vital for concise knowledge organization.

5.3 Test-time Scaling

Inspired by OpenAI o1 (Jaech et al., 2024)'s observation, our framework enables dynamic test-time compute scaling without model weight updates. We investigate two scaling dimensions: (1) *top-k scaling* where we vary the number of retrieved passages $k \in [0, 5]$, and (2) *expand-iter scaling* which controls document expansion iterations $iter \in [0, 5]$, With their impacts demonstrated in Figure 5.

Top-k Scaling. As shown in Figure 5 *left*, increasing k consistently improves both retrieval and answer correctness, as expanding the search space enhances the likelihood of capturing relevant documents. While standard *Retrieve-and-Read* ex-
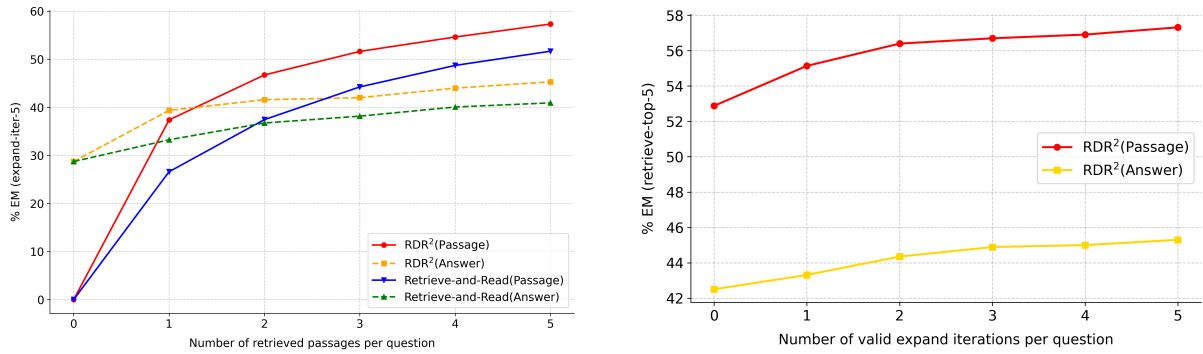
Figure 5: Scaling test-time compute on ASQA for RDR^2 framework. *Left*: *top-k scaling*. *Right*: *expand-iter scaling*. Exact Match (EM) is reported from both passage/answer-aspect.
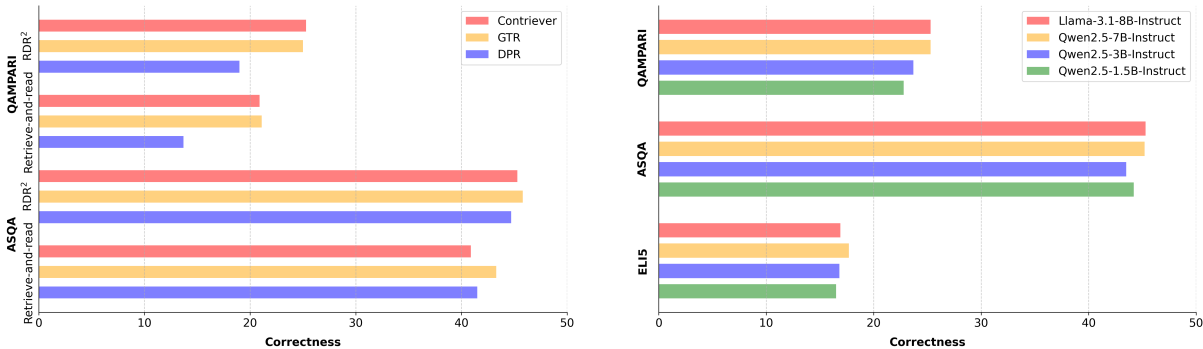

Figure 6: Robustness experiment across different datasets. *Left*: retriever robustness. *Right*: router robustness. Correctness metrics: F_1-5 for QAMPARI, Exact Match for ASQA and Claim Recall for ELI5.

hibits similar scaling trends, our framework maintains a consistent performance advantage. This suggests that structural awareness potentially enhances the benefits of retrieval test-time scaling.

Expand-*iter* Scaling. As shown in Figure 5 *right*, increasing expansion iterations yields consistent improvements in both passage utility and answer quality. Our controlled expansion mechanism introduces a novel RAG scaling paradigm, offering adjustable trade-offs between performance and computational cost - particularly valuable for applications with varying latency-accuracy requirements.

5.4 Robustness

Figure 4 demonstrates RDR^2's robustness to diverse readers and held-out datasets. We further investigate the retrievers and routers compatibility.

Retriever Robustness. We use off-the-shelf GTR (Ni et al., 2022) and DPR (Karpukhin et al., 2020) as the retriever. As shown in Figure 6 *left*, RDR^2 maintains stable performance with different retrievers across datasets, while standard *Retrieve-and-Read* exhibits performance fluctua-

tions, empirically validates that explicit structure perception enhances RAG's robustness to component variations.

Router Robustness. We fine-tuned routers based on Qwen2.5-Instruct (Qwen et al., 2025) series using the same protocol. As shown in Figure 6 *right*, experiments consistently validate our method's effectiveness across different model architectures and scales.

6 Conclusion

This work introduces RDR^2, a novel framework that explicitly incorporates document structure throughout the RAG process. Our approach dynamically navigates document structure trees using an LLM-based router, which jointly considers content relevance and hierarchical relationships to assemble optimal evidence. Comprehensive evaluations across five datasets demonstrate that document structure awareness brings significant and consistent gains to RAG systems, especially in scenarios requiring multi-document synthesis.

Limitations

We acknowledge three key limitations of this work: (1) While our routing mechanism effectively navigates intra-document hierarchies, it processes each document independently. The document count is determined by the initial top-k retrieval, potentially limiting inter-document knowledge integration. (2) The framework requires offline construction of Document Structure Trees (DSTs) for the entire datastore (approximately 20 minutes for Wikipedia in our experiment, with parallelization across 8 CPU cores). (3) The iterative routing process incurs computational overhead, which can be partially mitigated through controlled expansion iterations during inference.

Ethical Concerns

This study focuses on improving knowledge acquisition and utilization in RAG systems through document structure awareness. All data, models, and APIs used in our experiments are sourced from publicly available platforms to ensure transparency and reproducibility. We strictly adhere to ethical guidelines throughout the research process, guaranteeing that our work poses no harm to individuals or groups. Furthermore, we commit to avoiding any form of deception or misuse of information in both methodology and application.

References

Samuel Amouyal, Tomer Wolfson, Ohad Rubin, Ori Yoran, Jonathan Herzig, and Jonathan Berant. 2023. QAMPARI: A benchmark for open-domain questions with many answers. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 97–110, Singapore. Association for Computational Linguistics.

Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. 2023a. Retrieval-based language models and applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pages 41–46, Toronto, Canada. Association for Computational Linguistics.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023b. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, and 1 others. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Chunjing Gan, Dan Yang, Binbin Hu, Hanxiao Zhang, Siyuan Li, Ziqi Liu, Yue Shen, Lin Ju, Zhiqiang Zhang, Jinjie Gu, and 1 others. 2024. Similarity is not all you need: Endowing retrieval augmented generation with multi layered thoughts. *arXiv preprint arXiv:2405.19893*.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488, Singapore. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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

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

Lei Huang, Xiaocheng Feng, Weitao Ma, Yuxuan Gu, Weihong Zhong, Xiachong Feng, Weijiang Yu, Weihua Peng, Duyu Tang, Dandan Tu, and Bing Qin. 2024. Learning fine-grained grounded citations for attributed large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14095–14113, Bangkok, Thailand. Association for Computational Linguistics.

Shayekh Bin Islam, Md Asib Rahman, K S M Toza-mmel Hossain, Enamul Hoque, Shafiq Joty, and Md Rizwan Parvez. 2024. Open-RAG: Enhanced retrieval augmented reasoning with open-source large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14231–14244, Miami, Florida, USA. Association for Computational Linguistics.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

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

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Palak Jain, Livio Soares, and Tom Kwiatkowski. 2023. 1-PAGER: One pass answer generation and evidence retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14529–14543, Singapore. Association for Computational Linguistics.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38.

Zhengbao Jiang, Luyu Gao, Zhiruo Wang, Jun Araki, Haibo Ding, Jamie Callan, and Graham Neubig. 2022. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2336–2349, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. 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. Association for Computational Linguistics.

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

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4940–4957, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Richard James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, and 1 others. 2023. Ra-dit: Retrieval-augmented dual instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

LINHAO LUO, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2024. Reasoning on graphs: Faithful and interpretable large language model reasoning.

In *The Twelfth International Conference on Learning Representations*.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

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. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8371–8384, Mexico City, Mexico. Association for Computational Linguistics.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Raghuveer Thirukovalluru, Yukun Huang, and Bhuwan Dhingra. 2024. Atomic self-consistency for better long form generations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12681–12694, Miami, Florida, USA. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Liang Wang, Haonan Chen, Nan Yang, Xiaolong Huang, Zhicheng Dou, and Furu Wei. 2025. Chain-of-retrieval augmented generation. *arXiv preprint arXiv:2501.14342*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

Yuan Xia, Jingbo Zhou, Zhenhui Shi, Jun Chen, and Haifeng Huang. 2025. Improving retrieval augmented language model with self-reasoning. In *Pro-*

ceedings of the AAAI conference on artificial intelligence, volume 39, pages 25534–25542.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

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

Jihao Zhao, Zhiyuan Ji, Yuchen Feng, Pengnian Qi, Simin Niu, Bo Tang, Feiyu Xiong, and Zhiyu Li. 2024. Meta-chunking: Learning text segmentation and semantic completion via logical perception. *arXiv preprint arXiv:2410.12788*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024. Language agent tree search unifies reasoning, acting, and planning in language models. In *Forty-first International Conference on Machine Learning*.

A   Implementation Details

A.1   Dataset curation

	ANS	EXP	REF	Total
Train	14,822	3,793	5,212	23,827
Test	287	90	123	500

Table 3: Routing Dataset.

As shown in Table 3, the routing dataset consists of 23,827 training samples, including 14,822 [ANS] instances, 3,793 [EXP] instances, and 5,212 [REF] instances.

For the construction of the routing dataset, we begin by sampling queries from the ASQA training set and feed them into the retriever to obtain the top-k relevant text chunks. Each retrieved chunk is then aligned with the original *Document*

Structure Tree by applying the Levenshtein Distance algorithm. Specifically, a sliding-window strategy with a stride of one character is employed to locate the most similar spans within the structure tree. Once the mapping between a retrieved chunk and its corresponding content node is established, we apply the *RST Derivation Algorithm* (Algorithm 1) to systematically traverse the tree and retain all siblings, ancestors, and descendants of the mapped nodes that share the *content type* attribute. This procedure yields the corresponding *Retrieval Subtrees*. Finally, the DeepSeek-V3 API is leveraged to generate single-turn routing outputs conditioned on the given queries and their associated subtrees.

Algorithm 1 RST Derivation

Require: DST, $Lighted\ nodes$
1: **function** LIGHTNODES($Tree$, $Nodes$)
2:     **for** each $node \in Nodes$ **do**
3:         $siblings \leftarrow$ GETSIBLINGS($Tree$, $node$)   ▷ Acquiring necessary sibling nodes
4:         **for** each $sibling \in siblings$ **do**
5:             **if** $sibling$.type = "content" **then**
6:                 $sibling$.lighted $\leftarrow$ True
7:             **end if**
8:         **end for**
9:
10:         $current \leftarrow node$
11:         **while** $current$.parent $\neq \emptyset$ **do**
12:             $current \leftarrow current$.parent
13:             **if** $current$.type = "structure" **then**
14:                 **break**   ▷ Acquiring necessary upper ancestor nodes
15:             **end if**
16:             $current$.lighted $\leftarrow$ True
17:         **end while**
18:
19:         **for** each $sibling \in siblings$ **do**
20:             **if** $sibling$.type = "content" **then**
21:                 LIGHTDESCENDANTS($Tree$, $sibling$)   ▷ Acquiring necessary lower descendant nodes
22:             **end if**
23:         **end for**
24:     **end for**
25: **end function**

A.2   Training details

We choose Llama-3.1-8B-Instruct as the backbone of the routing model and employ LoRA for effi-

cient fine-tuning. Specifically, we set lora_rank as 8, lora_alpha as 16, gradient accumulated batch size as 8, learning rate as 1e-5 and epoch as 5. We also compare different training settings, as shown in Table 4, and finally select the model based on instruct model with tag format prompt.

B More Experiments

B.1 Main results

As shown in Table 5 and Table 6, we report full results of our main experiment. We can observe that:

1) With different backbone models, regardless of their openness or parameter scale, our framework consistently outperforms baseline methods across all evaluation metrics.

2) Compared to state-of-the-art approaches, our framework demonstrates superior performance on most metrics.

3) Furthermore, our framework significantly narrows the performance gap between open-source and proprietary models.

4) By learning document routing capabilities, our framework exhibits strong generalization ability on factual reasoning question answering tasks.

B.2 Ablation Study

Full results of the ablation study are shown in Table 7. To evaluate the end-to-end ranking correctness of the retrieval process, we propose the **Inverse Information Rank Score** Score_{psg}. Given the set of retrieved passages $C_{re} = \{c_{re}^{(i)}\}_{i=1}^{\text{top}_k}$ and the set of reference short answers A, the score is defined as follows.

$$\text{Score}_{psg} = \frac{\sum_{i=1:|C_{re}|} \frac{1}{i} \cdot \text{EM}(c_{re}^{(i)}, A)}{|C_{re}|} \quad (10)$$

This metric models the gain of correctness information with a position-based decay, which aligns with the tendency of both retrieval and generation modules to favor top-ranked results.

B.3 Test-time Scaling

We report statistics of test-time scaling in Table 8 and Table 9, including Top-k and Expand-*iter* scaling.

B.4 Chunking Comparison

We conduct a comparative experiment with Meta-Chunking (Zhao et al., 2024) on ASQA. Specifically, we applied Meta-Chunking to the retrieved documents with *target_size* = 100 to ensure comparable chunking sizes and *threshold* = 0 for perplexity chunking. For a fair comparison, we carefully adapted Meta-Chunking to our setting, with an edit-distance constraint calculated against the original chunking to mitigate potential mismatch introduced by different retrievers. The results are shown in the Table 10 (with Llama3.1-8B-Instruct based router and reader).

B.5 Short-form QA Performance

We conduct experiments on two short-form datasets, HotpotQA and TriviaQA. As shown in Table 11 ,compared to standard RAG, our method achieves consistent improvements: +8.0% EM on HotpotQA and +5.0% EM on TriviaQA. For reference, the correctness improvements on ASQA, QAMPARI and ELI5 are +10.8%, +21.1% and +3.7% respectively. These improvements can be attributed to different underlying mechanisms. For HotpotQA, document structures facilitate multi-hop reasoning by explicitly modeling relationships between passages. For TriviaQA, where answers often rely on single passages, the gain likely stems from better context organization via structural awareness.

B.6 Router Generalization Experiments

As shown in Table 12, we finetuned Qwen2.5-7B-Instruct as the router, observing comparable performance within each datasets. Moreover, experiments with routers finetuned on Qwen2.5 series (1.5B/3B/7B) consistently validate our method's effectiveness across different model scales.

B.7 Further Analysis

We further provide details of our method from the perspective of time delay, computing budget and hierarchy modeling.

1) The offline DST construction is a deliberate design choice to achieve real-time retrieval efficiency, analogous to how standard RAG pipelines require offline processing steps like FAISS index building. In our experiments, constructing DSTs for the entire Wikipedia dump (5.82M documents) takes approximately 20 minutes with parallelization across 8 CPU cores.

2) We evaluate the computing efficiency on ASQA dataset. Specifically, with single-turn routing, our method achieves +3.9% EM over standard RAG while adding minimal computational cost (+0.779k / 0.017k input/output tokens). This

Prompt	Post-processing	Model	Train	Epoch	ANS-F1	EXP-PRE	REF-ACC	XPL-AVG	COL-RATE
enclose	√	base	full	7.0	**86.3**	46.9	82.9	0.6	**0.0**
enclose	√	inst	full	7.0	86.1	50.3	83.7	0.6	**0.0**
enclose	√	inst	lora	4.5	84.0	55.7	81.3	**0.4**	**0.0**
tag	√	inst	lora	3.5	83.0	**57.1**	**87.0**	**0.4**	**0.0**
tag	×	inst	lora	5.0	84.4	51.1	77.2	7.6	0.4

Table 4: Comparison of different training settings. We evaluate performance of fine-tuned routing models on the curated test set. Specifically, ANS-F1 denotes the f1 score of [ANS] action, EXP-PRE indicates the precision of [EXP] action, REF-ACC represents the accuracy of the [REF] action, XPL-AVG is the percentage of expelled output, and COL-RATE is the rate of collapsed output. Enclose and tag prompt represent the format of "[expand]" and "<expand></expand>", respectively.
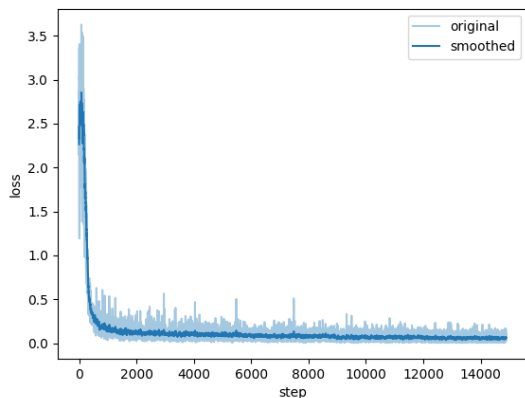
Figure 7: Training loss curve of routing model.

Figure 8: Validation set performance of routing model.

low latency is primarily attributed to our RST design, concise router input/output format and parallelizable top-k retrieval/routing, before a single-turn final answer generation. Moreover, when employing multi-turn expansion, EM gains increase significantly to +10.8% with modest overhead (+2.031k / 0.041k input/output tokens). Crucially, this performance-efficiency trade-off is tunable via the expand-iter hyperparameter, requiring no retraining.

3) For the hierarchy modeling, the Document Structure Tree (DST) can naturally handle both documents with rich structure and documents without a clear hierarchy, representing them as simple trees (e.g., single-level structures for flat documents). Our supplementary analysis shows that such cases are relatively rare in practice: on ASQA with Wikipedia documents, only 6.5% of retrieved documents are shallow/flat (depth 2), where our method still yields a +0.4 EM improvement over standard RAG. As shown in Table 13,

for the majority of documents (85.8% with depth 2-4), we observe substantial improvements (+4.1 EM). For deeply structured documents (depth > 4), the gains are even more significant (+10.8 EM).

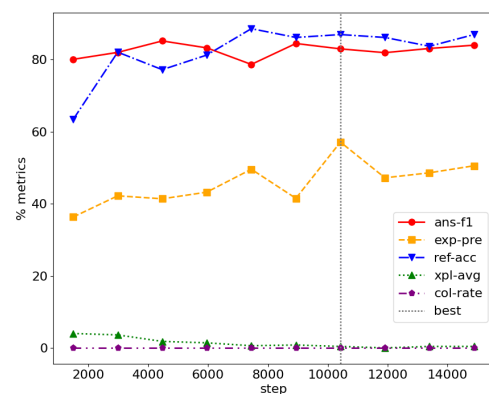| Methods	ASQA				
	EM(HIT)	**D-F_1**	**ROUGE**	**MAUVE**	**LEN**
Reader based on text-davinci-003					
No-retrieval	33.8	24.2	33.3	-	-
Retrieve-and-Read	40.0	27.1	34.0	-	-
FLARE	<u>41.3</u>	<u>28.2</u>	<u>34.3</u>	-	-
Reader based on ChatGPT					
No-retrieval	34.1(9.7)	27.4	35.7	18.2	57.5
Retrieve-and-Read	42.8(16.1)	34.4	38.0	57.0	51.1
ASC-F	45.0	31.9	-	41.3	106.7
ASC	44.1	32.2	-	47.0	101.2
RDR^2(Ours)	<u>46.1(18.6)</u>	<u>37.1</u>	<u>38.5</u>	<u>70.6</u>	49.1
Reader based on GPT-4o					
No-retrieval	41.4(13.7)	33.9	36.2	23.3	58.8
Retrieve-and-Read	47.0(19.1)	36.5	<u>38.4</u>	39.9	68.4
RDR^2(Ours)	<u>48.2(21.0)</u>	<u>39.0</u>	<u>38.4</u>	<u>48.3</u>	63.8
Reader based on DeepSeek-V3					
No-retrieval	43.0(16.7)	33.1	36.3	21.9	69.2
Retrieve-and-Read	48.8(21.9)	37.4	37.5	36.7	74.2
RDR^2(Ours)	**50.8(23.2)**	**39.8**	<u>37.8</u>	<u>37.3</u>	68.9
Reader based on Llama-2-13b					
No-retrieval	24.7(6.5)	19.3	35.1	13.4	65.9
Retrieve-and-Read	36.5(13.5)	26.9	39.2	31.4	61.2
Self-Reasoning(FT)	35.2	-	-	-	-
Self-RAG(FT)	31.7(8.4)	26.4	37.0	71.6	27.0
Self-RAG(FT)*	37.5(14.9)	27.5	39.2	77.7	69.9
Open-RAG(FT)	36.3	-	38.1	**80.0**	-
Open-RAG(FT)*	39.9(14.5)	24.0	**40.4**	17.2	83.7
Front	41.5	-	38.6	76.1	57.6
RDR^2(Ours)	<u>41.7(16.9)</u>	<u>31.6</u>	39.2	61.2	69.6
Reader based on Llama-3.1-8b					
No-retrieval	28.7(7.5)	22.0	34.7	40.7	65.2
Retrieve-and-Read	40.9(15.9)	30.9	37.9	73.6	69.2
RDR^2(Ours)	<u>45.3(18.7)</u>	<u>34.9</u>	<u>38.2</u>	<u>79.2</u>	71.3

Table 5: Main results on ASQA dataset. We report full results of different API and open-source models, together with results of no-retrieval and retrieve-and-read baselines. **Bold** and <u>Underline</u> denote the best overall and in-category results, respectively. FT refers to methods finetuned on the corresponding training set. * marks the results from our reproduction.

Methods	QAMPARI			ELI5		
	F_1-5(F_1)	REC-5(REC)	PRE	CLAIM	MAUVE	LEN
Reader based on ChatGPT						
No-retrieval	17.7(12.5)	18.5(10.6)	20.8	<u>23.6</u>	14.4	145.3
Retrieve-and-Read	22.2(15.7)	22.1(13.4)	27.9	22.3	12.2	141.5
ASC-F	18.8(15.7)	**45.0(29.8)**	13.4	22.2	<u>22.7</u>	172.7
ASC	26.2(19.5)	33.0(20.5)	23.0	21.4	21.3	163.6
RDR^2(Ours)	<u>26.4(19.8)</u>	29.0(18.7)	<u>30.9</u>	23.3	14.4	155.2
Reader based on GPT-4o						
No-retrieval	26.0(19.3)	28.7(17.6)	30.5	<u>26.5</u>	<u>20.2</u>	158.4
Retrieve-and-Read	23.7(17.0)	23.2(14.7)	31.0	24.5	17.9	154.7
RDR^2(Ours)	**28.4**(<u>21.4</u>)	<u>30.8(20.3)</u>	**34.8**	25.3	16.9	165.3
Reader based on DeepSeek-V3						
No-retrieval	23.4(18.4)	28.7(18.8)	23.3	26.3	<u>15.6</u>	137.4
Retrieve-and-Read	23.2(17.1)	24.6(15.7)	27.3	26.6	14.9	132.4
RDR^2(Ours)	<u>27.8(**21.7**)</u>	<u>32.1(21.6)</u>	<u>31.1</u>	**27.4**	13.2	152.3
Reader based on Llama-2-13b						
No-retrieval	14.9(10.3)	16.4(9.0)	14.3	14.7	21.9	140.2
Retrieve-and-Read	21.0(14.7)	22.0(12.9)	21.6	14.9	20.8	141.2
SELF-RAG	-	1.9	1.3	6.1	-	-
SELF-RAG*	6.5(4.9)	9.0(5.5)	6.4	11.8	**42.8**	81.9
OPEN-RAG*	2.5(1.9)	3.9(2.3)	2.3	11.9	19.1	129.2
FRONT	-	11.9	22.6	9.3	34.4	75.1
RDR^2(Ours)	<u>23.2(16.7)</u>	<u>24.3(14.9)</u>	<u>25.0</u>	<u>15.4</u>	23.9	148.3
Reader based on Llama-3.1-8b						
No-retrieval	13.8(10.3)	19.3(11.1)	13.1	16.0	18.8	139.5
Retrieve-and-Read	20.9(15.1)	23.6(14.3)	22.9	16.3	<u>21.6</u>	141.9
RDR^2(Ours)	<u>25.3(19.5)</u>	<u>32.3(21.1)</u>	<u>25.7</u>	<u>16.9</u>	20.3	141.6

Table 6: Main results on QAMPARI and ELI5 datasets. We report full results of different API and open-source models, together with results of no-retrieval and retrieve-and-read baselines. **Bold** and <u>Underline</u> denote the best overall and in-category results, respectively. * marks the results from our reproduction.

Method	P-EM(P-HIT)	P-SCORE	P-LEN	EM(HIT)	D-F_1	LEN
RDR^2(Ours)	57.3(34.2)	12.7	104.2	**45.3(18.7)**	**38.2**	71.3
w/o router	51.7(28.3)	10.2	100.0	40.9(15.9)	30.9	69.2
w/o structure	49.8(28.0)	10.6	67.5	41.3(15.0)	32.5	71.0
w/o similarity	54.8(32.7)	11.8	100.9	43.9(17.8)	33.2	72.3
w/o content	54.2(31.3)	11.7	93.9	43.7(18.0)	34.0	70.0
w/o [expand]	52.9(30.8)	11.5	81.7	42.5(16.1)	32.5	71.9
w/o [refuse]	**61.2(37.0)**	**13.4**	176.3	42.9(16.4)	32.8	70.7

Table 7: Ablation results of RDR^2(ASQA).

Methods	EM(HIT)	D-F_1	ROUGE	MAUVE	LEN
Retrieve-and-Read					
top-0	28.7(7.5)	22.0	34.7	40.7	65.2
top-1	33.2(11.8)	25.4	35.1	68.1	65.2
top-2	36.7(13.2)	28.2	36.8	70.2	66.6
top-3	38.2(14.0)	29.5	37.3	75.8	69.7
top-4	40.1(14.6)	30.7	37.7	74.0	70.5
top-5	40.9(15.9)	30.9	37.9	73.6	69.2
RDR^2(Ours)					
top-0	28.7(7.5)	22.0	34.7	40.7	65.2
top-1	39.4(15.1)	30.2	36.0	69.1	70.5
top-2	41.6(16.5)	32.6	37.7	71.8	69.6
top-3	42.0(16.5)	32.9	37.8	77.7	68.6
top-4	44.0(17.1)	33.9	38.2	76.2	69.4
top-5	45.3(18.7)	34.9	38.2	79.2	71.3

Table 8: Statistics of Top-*k* scaling.

Methods	P-EM(P-HIT)	P-SCORE	P-LEN	EM(HIT)	D-F_1	ROUGE	MAUVE	LEN
RDR^2(Ours)								
iter-0	52.9(30.8)	11.5	81.7	42.5(16.1)	32.5	37.7	76.7	71.9
iter-1	55.1(32.7)	12.1	95.5	43.3(17.1)	33.3	37.9	75.9	72.0
iter-2	56.4(33.2)	12.5	95.4	44.3(18.9)	34.5	38.1	78.7	69.3
iter-3	56.7(33.5)	12.6	98.9	44.9(19.6)	35.1	38.2	78.9	68.9
iter-4	56.9(33.7)	12.6	100.1	45.0(18.6)	34.8	38.2	76.0	71.1
iter-5	57.3(34.2)	12.7	104.2	45.3(18.7)	34.9	38.2	79.2	71.3

Table 9: Statistics of Expand-*iter* scaling.

Methods	Chunking	ASQA				
		EM(HIT)	D-F_1	ROUGE	MAUVE	LEN
No-retrieval	-	28.7(7.5)	22.0	34.7	40.7	65.2
Retrieve-and-Read	Fix-chunking	40.9(15.9)	30.9	37.9	73.6	69.2
Retrieve-and-Read	Meta-Chunking (MSP)	41.3(16.4)	30.0	37.0	76.1	74.3
Retrieve-and-Read	Meta-Chunking (PPL)	40.8(15.2)	29.4	37.1	70.6	75.4
RDR^2	-	**45.3(18.7)**	**34.9**	**38.2**	**79.2**	71.3

Table 10: Comparison with Meta-Chunking on ASQA dataset.

Method	Router (SFT)	HotpotQA		TriviaQA	
		EM	F1	EM	F1
No-Retrieval	-	25.6	28.2	60.3	60.1
Retrieve-and-Read	-	37.5	41.8	68.6	69.0
RDR^2	Llama-3.1-8B-Instruct	**40.5**	44.3	**72.0**	**72.6**
RDR^2	Qwen2.5-7B-Instruct	39.8	**44.7**	69.9	70.9

Table 11: Results on HotpotQA and TriviaQA datasets.

Router (SFT)	Reader	ASQA				QAMPARI			ELI5	
		EM(HIT)	D-F$_1$	R-L	Mau	F1-5(F1)	Rec-5(Rec)	Pre	Cla	Mau
Llama-3.1-8B-Inst	Llama-3.1-8B-Inst	**45.3**(18.7)	34.9	38.2	**79.2**	**25.3(19.5)**	**32.3(21.1)**	**25.7**	16.9	20.3
Qwen2.5-7B-Inst	Llama-3.1-8B-Inst	45.2(**18.9**)	**35.1**	**38.7**	75.4	**25.3**(19.3)	31.2(19.9)	**25.7**	**17.7**	**24.5**
Qwen2.5-3B-Inst	Llama-3.1-8B-Inst	43.5(18.6)	33.3	38.0	73.7	23.7(17.8)	28.7(18.1)	24.7	16.8	23.1
Qwen2.5-1.5B-Inst	Llama-3.1-8B-Inst	44.2(17.6)	34.1	38.4	76.1	22.8(16.9)	26.5(16.5)	24.7	16.5	22.2
Llama-3.1-8B-Inst	DeepSeek-V3	50.8(23.2)	**39.8**	37.8	37.3	**27.8(21.7)**	**32.1(21.6)**	**31.1**	**27.4**	13.2
Qwen2.5-7B-Inst	DeepSeek-V3	50.7(**23.4**)	39.4	**38.1**	44.5	26.8(20.4)	30.2(19.6)	30.4	27.0	**18.6**

Table 12: Comparison of different routers on ASQA, QAMPARI, and ELI5 datasets.

DST Depth	Num	Standard RAG (EM)	RDR^2 (EM)	ΔEM
[1,2]	62	52.6	53.0	+0.4
(2,4]	813	49.8	44.9	+4.1
(4,7]	73	32.8	43.6	+10.8

Table 13: Analysis of document depth and performance.

C Prompts

We show the detailed prompts for data curation, routing and inference as follows:

Prompt C.1: Training data curation prompt

```
You are an expert in reading comprehension tasked with identifying relevant paragraphs from a
document tree to answer a question. Follow these steps carefully:

1. Strict Relevance Assessment:
* First determine if the document's root heading is fundamentally relevant to the question.
* If the document is clearly about a different topic, immediately return "Cannot answer".
* Only proceed if the document is relevant or potentially relevant to the question.

2. Comprehensive Answer Extraction:
* For expanded paragraphs (visible content):
  - Tag as "answer" ONLY if the paragraph DIRECTLY and COMPLETELY answers the question.
  - If multiple paragraphs together provide a complete answer, tag ALL relevant ones.
  - When paragraphs contain conflicting or supplementary information, include all that are relevant.

3. Collapsed Heading Expansion:
* If any unexpanded nodes might contain information that can answer the question? Tag as "expand"
when ANY of these are true:
  - The heading contains synonyms or standard terminology related to the question.
  - The section appears in the expected position within a standardized document structure.
  - Expanded sibling sections under the same parent contain answers.

4. Output Requirements:
* Strictly use this JSON format:
  [
    {
      "id": [integer],
      "tag": "answer"|"expand",
      "explanation": "[concise rationale]"
    }
  ]
  OR "Cannot answer".
* Never include irrelevant paragraphs just because they mention similar keywords.
* For multi-part answers, include ALL relevant paragraphs.
* If no paragraphs meet the strict criteria, return "Cannot answer".

## Question
{question}

## Document
{context}
```

Prompt C.2: Routing module prompt

```
You are asked to identify relevant nodes in a document tree that can answer the given question. Use
[ANSWER] if a paragraph directly contributes to answering the question. Use [EXPAND] if a collapsed
heading might contain information that can answer the question. If neither exists, reply exactly
"Cannot answer".
## Question
{question}

## Document
{context}

## Response
```

Prompt C.3: Reader prompt for TriviaQA HotpotQA

```
Instruction: Provide one accurate answer for the given question. Do not explain yourself or output
anything else.
## Paragraph
{paragraph}

## Question
{question}

## Response
```

Prompt C.4: Reader prompt for QAMPARI

```
Instruction: Provide a list of accurate answers for the given question. Separate answers by commas.
Do not explain yourself or output anything else.
## Paragraph
{paragraph}

## Question
{question}

## Response
```

Prompt C.5: Reader prompt for ASQA ELI5

```
Instruction: Write an accurate, engaging, and concise answer for the given question. Use an unbiased
and journalistic tone.
## Paragraph
{paragraph}

## Question
{question}

## Response
```

D Demonstrations of RDR^2

We show a complete demonstration of our **RDR^2** as follows, including comparison of generation and retrieval stage and detailed routing actions.

User Question	What is the tallest ride at six flags over texas?
Short Answers	*The Titan*, *The New Texas Giant*, *The Texas Giant*, *Texas SkyScreamer*, *Superman: Tower of Power*
Long Answer	The Six Flags Over Texas theme park has a number of different rides and roller coasters. The park's tallest swing ride is the *Texas SkyScreamer* at 400 feet and tallest triple drop tower ride is *Superman: Tower of Power* at 325 feet. The tallest roller coaster at the park in 2001 was *The Titan* at 245 feet. The park's tallest wooden roller coaster in 2011 was *The New Texas Giant* at 153 feet, while its tallest one in 1990 was *The Texas Giant* at 143 feet.
No-Retrieval Output	At Six Flags Over Texas, thrill-seekers can experience the park's tallest ride, the 'Mr. Freeze: Reverse Blast'. However, more recently, it has been surpassed by the 'Dare Devil Dive' in the 'Thrill Zone' area of the park, but that is no longer there.
RAG Output	According to the information provided, Six Flags Over Texas has had several record-breaking attractions over the years. In 2003, the park opened the *Superman Tower of Power*, which was the tallest ride of its kind in the world at the time of its opening. However, it does not specify if it is still the tallest ride in the park.
RDR² Output	According to the information provided, the tallest ride at Six Flags Over Texas is the *Texas SkyScreamer*, which stands at a height of 400 feet. In 2003, the park opened the *Superman Tower of Power*, which was the tallest ride of its kind in the world at the time. It's also worth noting that the park has had several record-breaking roller coasters, including *the Titan*, which is the tallest, fastest, and longest coaster in Texas.

Table 14: End-to-end comparison between three frameworks.

Top-3 Retrieved Chunks

Six Flags Over Texas	bringing back the past when they reopened Casa Magnetica. In 2003 Six Flags Over Texas opened the ==*Superman Tower of Power*==. This was the tallest ride of its kind in the world at the time of its opening. [...] For 2007 and 2008, Six Flags Over Texas was home to "Cirque',
Six Flags Over Texas	end of the decade, Six Flags Over Texas had added ten roller coasters to its list of attractions. During the first decade of the 21st century, Looney Tunes USA was restructured. In 2001, the park introduced its tallest, fastest, longest roller coaster, ==*Titan*==. [...] The park has also made steps toward
Titan (roller coaster)	==*Titan*== is a steel hyper coaster located at Six Flags Over Texas in Arlington, Texas. Unlike most hypercoasters, Titan is a combination of an out and back roller coaster and a twister roller coaster. It stands at 245 feet and contains a 255 drop at 85 miles per hour. It is the tallest, fastest, and longest coaster in Texas. [...] In August 2000, Six

Corresponding Routed Chunks

Six Flags Over Texas	During the first decade of the 21st century, Looney Tunes USA was restructured. In 2001, the park introduced its tallest, fastest, longest roller coaster, Titan. [...] In 2003 Six Flags Over Texas opened the ==*Superman Tower of Power*==. This was the tallest ride of its kind in the world at the time of its opening. [...] * Tallest Roller Coaster in Texas - ==*Titan*== (245ft) * Tallest swing ride in the world ==*Texas Skyscreamer*== (400ft) (2013)
Titan (roller coaster)	==*Titan*== is a steel hyper coaster located at Six Flags Over Texas in Arlington, Texas. Unlike most hypercoasters, Titan is a combination of an out and back roller coaster and a twister roller coaster. It stands at 245 feet and contains a 255 drop at 85 miles per hour. It is the tallest, fastest, and longest coaster in Texas.

Table 15: Comparison between retrieved and routed chunks.

Document Structure Tree

-1: Six Flags Over Texas
   0: ==Introduction==
     1: Six Flags Over Texas is a 212-acre (86 ha) theme park located in Arlington, Texas, east of Fort Worth and about 15 miles (24km) west of Dallas. [...]
     2: The park is managed by the Six Flags Entertainment Corp., which also owns 53.1% interest of the Texas Limited Partnership that owns the park. [...]
   3: ==History==
     4: ===Initial planning and construction===
     [...]
     16: ===1990s===
     17: The 1990s was a rather rough decade in comparison from decades past. The decade started off with a bang when Six Flags Over Texas introduced the Texas Giant roller coaster. [...]
     18: ===2000s===
     19: During the first decade of the 21st century, Looney Tunes USA was restructured. In 2001, the park introduced its tallest, fastest, longest roller coaster, *Titan* . [...] In 2003 Six Flags Over Texas opened the *Superman Tower of Power* . This was the tallest ride of its kind in the world at the time of its opening. [...]
     20: ===2010s===
   29: ==Firsts, bests, and other records==
     30: ===Firsts and ones of a kind===
     40: ===Records===
      41: * Tallest Roller Coaster in Texas - *Titan* (245ft)
      42: * Fastest Roller Coaster in Texas - Titan (85mph)
      43: * Largest Land Based Oil Derrick - Oil Derrick (300ft)
      44: * Tallest swing ride in the world *Texas Skyscreamer* (400ft) (2013)
     45: ===Awards===
   48: ==Events==
   54: ==Areas and attractions==
     56: ===Star Mall===
     [...]
     157: ===Tower===
   168: ==Former Attractions==

Routing Actions

Light content node *17* from retrieved passages.
1 **[EXPAND]** *0*
Light content node *1*, *2* from expand action.
2 **[REFUSE]**
Light content node *19* from retrieved passages.
3 **[ANSWER]** *19* **[EXPAND]** *40*
Light content node *41*, *42*, *43*, *44* from expand action.
4 **[ANSWER]** *41*, *44*

Routing Passages: *19*, *41*, *44*

Table 16: Demonstration of routing actions.