

Verbal-R3: Verbal Reranker as the Missing Bridge between Retrieval and Reasoning

Sangkwon Park^{1*}, Donghun Kang^{1*}, Jisoo Mok^{4†}, Sungho Yoon^{1,2,3†}

¹Department of Electrical and Computer Engineering, Seoul National University

²Interdisciplinary Program in Artificial Intelligence, Seoul National University

³AIIS, ASRI, INMC, and ISRC, Seoul National University ⁴DGIST

{tkdrnjs0621, 0k9d0h1, sryoon}@snu.ac.kr, jmok@dgist.ac.kr

Abstract

The conventional Retrieval-Augmented Generation (RAG) paradigm of injecting raw retrieved texts into the Large Language Model (LLM)’s context often results in suboptimal integration of retrieved information. This paper proposes to bridge retrieval results and the LLM’s reasoning ability through Verbal Annotations, analytic narratives that explicitly articulate the logical connection between a search query and retrieved contexts. Our empirical investigation reveals the potential of Verbal Annotations to substantially enhance the LLM’s ability to generate accurate, contextually-grounded responses. Motivated by this finding, we introduce Verbal-R3, a novel agentic RAG framework that consists of a Generator and a Verbal Reranker. The Generator performs iterative retrieval and reasoning, while the Verbal Reranker returns relevance scores and Verbal Annotations to guide the reasoning and answering process of the Generator. The inference process of Verbal-R3 is further refined through relevance-guided test-time scaling, which efficiently allocates test-time compute for effective trajectory expansion. Verbal-R3 achieves state-of-the-art performance on complex Question Answering benchmarks, validating the effectiveness of the proposed framework.¹

1 Introduction

The Large Language Models’ (LLMs) (Hurst et al., 2024; Yang et al., 2025) tendency to hallucinate and the inevitable constraints of fixed training cut-offs undermine the reliability and trustworthiness of their outputs (Huang et al., 2025). To mitigate such issues, Retrieval-Augmented Generation (RAG) has naturally emerged as a promising framework (Lewis et al., 2020). By enabling an explicit integration of additional or up-to-date knowledge

from external databases or sources into the LLMs’ generation process, RAG extends the applicability of LLMs to knowledge-intensive tasks.

The most naïve approach to RAG involves directly injecting raw retrieved texts into the LLMs’ context (Gao et al., 2023b). However, this prevailing paradigm in RAG remains suboptimal due to exposure bias, a mismatch between the pre-training data distribution and retrieved contexts, and the frequent inclusion of irrelevant, distracting information. Overcoming this technical bottleneck necessitates an intermediate module capable of transforming raw retrieval results (Li et al., 2025b).

To bridge this gap, we propose to adopt *Verbal Annotations*, an analytic narrative that verbalizes the logical connection between the query and the retrieved contexts. As shown in Figure 1, Verbal Annotations go beyond merely paraphrasing or summarizing a retrieved document; they state the specific alignment between the query and the document, providing a concrete logical bridge that the LLM can work with. To elucidate the significance of Verbal Annotations, we conduct an experimental investigation into how various methods of formatting and feeding retrieval results influence the accuracy of the final answer generated by an LLM. Interestingly, simply paraphrasing retrieved texts to match the stylistic patterns preferred by an LLM struggles to encourage an effective integration of retrieval results. On the contrary, Verbal Annotations enhance the correctness of the final answer by stating the relevance of the retrieved information and removing irrelevant noise.

Inspired by the effectiveness of Verbal Annotations, this work introduces the concept of a *Verbal Reranker*: an auxiliary LLM capable of evaluating the relevance of retrieved documents to the query and transforming them into a more digestible form, *i.e.*, Verbal Annotations. Our proposed two-part RAG system, dubbed Verbal-R3 (Retrieve-Rerank-Reason), is comprised of a Generator and

* Equal Contribution

† Corresponding Authors

¹<https://github.com/0k9d0h1/VerbalR3>

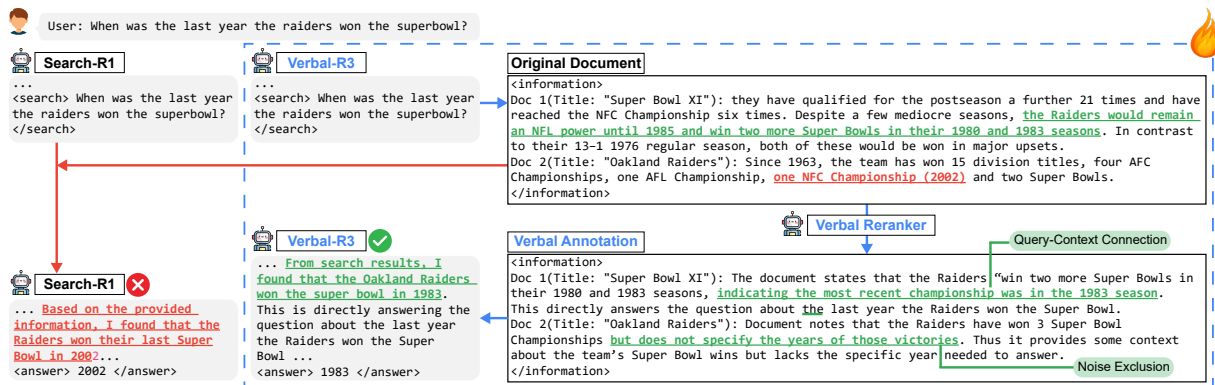


Figure 1: Comparison of Search-R1 vs. Verbal-R3 inference procedures. The original document contains a large amount of extraneous information, which makes Search-R1 hallucinate and produce an incorrect answer. In contrast, Verbal-R3 successfully exploits Verbal Annotations to distinguish relevant information, allowing the Generator to avoid hallucinations and generate a correct answer.

a Verbal Reranker LLMs. The Generator identifies information required to answer the user’s question and produces a retrieval query accordingly. The Verbal Reranker then scores and reorders retrieved documents, while additionally rewriting each document into a verbalized form that states its relevance. This process is repeated until the Generator determines that sufficient information has been obtained to produce the final answer.

The training pipeline of Verbal-R3 consists of Verbal Reranker distillation and Generator alignment. Because relying on a proprietary or large reasoning model, *e.g.*, GPT-OSS-120B (OpenAI et al., 2025), incurs a huge computational cost, we collect query-document-Verbal Annotation triplets from such a model and distill them to smaller models with 1.5B or 3B parameters through supervised fine-tuning. Afterwards, the Generator is aligned with the trained lightweight Verbal Reranker via Group Relative Policy Optimization (GRPO) (Shao et al., 2024). Additionally, at inference time, we newly incorporate relevance-guided test-time scaling that leverages the Verbal Reranker as a query evaluator for the Generator, enabling a computationally efficient yet effective trajectory expansion.

Extensive experiments demonstrate that our lightweight Verbal Reranker more effectively improves RAG performance than existing reranking solutions. Moreover, our Verbal-R3 framework surpasses previous state-of-the-art results across a diverse range of Question Answering benchmarks. Our contributions are largely three-fold:

- We reveal that Verbal Annotations, structured analytic narratives that map the query-context relationship, are essential for enhancing an

LLM’s ability to produce accurate responses in knowledge-intensive tasks.

- Building on the effectiveness of Verbal Annotations, we propose Verbal-R3, a novel two-agent RAG framework that interleaves a Generator for iterative retrieval and reasoning and a Verbal Reranker that generates Verbal Annotations to guide the Generator’s reasoning.
- We introduce relevance-guided test-time scaling to Verbal-R3 that dynamically assigns test-time computation to the most promising reasoning paths and queries.

2 Related Works

2.1 Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) was introduced to mitigate hallucination, overcome knowledge cutoff limitations, and improve robustness for domain-specific knowledge (Lewis et al., 2020; Gao et al., 2023b). Various approaches have been proposed to improve RAG performance, including designing effective retrievers (Izacard et al., 2021; Wang et al., 2022; Zhang et al., 2025) and enhancing their utility through query and document rewriting techniques (Ma et al., 2023a; Gao et al., 2023a; Li et al., 2025b).

However, a single retrieval step is often insufficient for complex queries requiring multi-hop reasoning. To address this limitation, agentic approaches that treat retrieval as a tool have emerged. These methods have been explored through both test-time approaches (Trivedi et al., 2023; Shao et al., 2023; Jiang et al., 2023; Press et al., 2023b)

Method	2Wiki	Bamboogle	HotpotQA	MuSiQue	NQ	PopQA	TriviaQA	Average
Search-R1 3B	38.06	32.75	37.50	14.69	43.49	43.98	60.75	38.75
Search-R1 3B + Paraphrase	37.38	32.05	35.42	13.94	41.94	43.11	60.59	37.78
Search-R1 3B + Verbal Annotation	38.65	40.40	39.58	16.01	45.71	45.83	67.28	41.92

Table 1: Analysis of how different context integration methods affect the Search-R1 performance on QA benchmarks.

and training-based methods (Asai et al., 2024; Yu et al., 2024). To address the scarcity of ground-truth reasoning trajectories, recent works have integrated reinforcement learning to optimize search-based reasoning models (Jin et al., 2025b; Zheng et al., 2025). While Search-o1 (Li et al., 2025b) explores an evaluation of retrieved documents within agentic retrieval, our Verbal Reranker distinguishes itself in that its outputs are structurally designed to capture logical query–context relationships.

2.2 Rerankers

Reranking serves as a critical refinement step in retrieval pipelines, where an initial set of candidate documents retrieved by a first-stage system is reordered to prioritize the most relevant items by modeling the joint relation between the query and each document (Gao et al., 2023b).

The advent of transformer-based architectures (Vaswani et al., 2017; Devlin et al., 2019) marked a significant shift in reranking capabilities. Cross-encoders and late-interaction models (Nogueira and Cho, 2019; Khattab and Zaharia, 2020) leveraged BERT-based encoders to produce relevance scores, while sequence-to-sequence models using T5 (Raffel et al., 2020) as their backbone reformulated reranking as a generation task (Nogueira et al., 2019, 2020; Zhuang et al., 2023; Yoon et al., 2024). Recent work has explored leveraging large language models for reranking through both zero-shot prompting (Sun et al., 2023; Ma et al., 2023b) and supervised fine-tuning (Pradeep et al., 2023a,b). Our work adopts a pointwise Verbal Reranker that generates natural language explanations alongside relevance assessments, thereby providing functionality beyond conventional rerankers.

3 Motivation

Verbal Annotation. The suboptimal performance of RAG systems is commonly attributed to exposure bias, the distributional shift between an LLM’s pre-training data and raw retrieved contexts it encounters at inference time (Ranzato et al., 2015; Arora et al., 2022). This chasm between retrieved information and the LLM’s reasoning capabilities is further deepened by the presence of

tangential noise within raw retrieval results. Building on this observation, we posit that *Verbal Annotations*, analytic statements that explicitly articulate the relevance of the retrieved contexts, can serve as a crucial bridge to improve the fidelity of the final answer. Unlike standard context injection, a Verbal Annotation does not merely present additional text; it actively interprets the retrieved context to distinguish relevant evidence from irrelevant noise, articulates precisely why specific segments support the query, and justifies the exclusion of non-pertinent parts. In doing so, a Verbal Annotation aligns the external data with the internalized logical requirements of the LLM.

Significance of Verbal Annotations. To assess the impact of Verbal Annotations, we conduct a comparative analysis using Search-R1 (Jin et al., 2025b) as the backbone LLM, evaluating three distinct methods of feeding retrieved contexts on Question Answering (QA) benchmarks:

- **Naïve Context Integration:** Directly injecting original retrieved passages into the LLM’s input context with the prompt in Figure A2.
- **Paraphrased Context:** Rewriting passages using an external module in the stylistic patterns preferred by LLMs, improving their readability without a logical commentary.
- **Verbal Annotations:** Incorporating an analytic narrative that verbalizes the logical connection between the query and the documents.

In our exploratory experiments, both the Paraphrased Context and Verbal Annotations are obtained by prompting the GPT-OSS-120B model (OpenAI et al., 2025) using the templates in Figures A3 and A4. The prompt template used to generate Verbal Annotations was deliberately refined to produce higher-quality annotations, and the refinement process is further detailed in Section 4.1. To complement Verbal Annotations, GPT-OSS-120B additionally assigns a numerical relevance score to each document through a point-wise reranking approach, which assesses each document separately.

Our results in Table 1 demonstrate that Paraphrased Context underperforms Naïve Context In-

Model	EM	F1	RA-R	RA-L	CUE-R	CUE-L
w/o VA	38.75	46.31	61.79	48.26	64.77	69.04
w/ VA	41.92	49.86	62.84	46.37	70.76	74.10

Table 2: Analysis of Search-R1 with and without Verbal Annotation (VA). RA denotes Retrieval Accuracy.

tegration, whereas employing Verbal Annotations significantly outperforms both baselines. This result suggests that simply smoothing the text distribution through paraphrasing may inadvertently filter out crucial details or introduce unintended hallucinations. In contrast, Verbal Annotations act as a rigorous filter that explicitly elaborates the relationship between the query and the document. By analyzing which parts of the document are relevant, they prevent the LLM from being distracted by noise, serving as a cognitive bridge rather than a superficial stylistic transfer.

Analysis of Verbal Annotations. To evidence that Verbal Annotations improve the correctness of the final answer by facilitating an effective incorporation of retrieval results, we explore the effect of Verbal Annotations through a new metric, Context Utilization Efficacy (CUE). CUE is computed by the conditional probability of the final answer being correct given that the retrieval result is accurate:

$$\text{CUE} = P(\text{Answer Correct} \mid \text{Retrieval Accurate}).$$

The Retrieval Accuracy (RA) can be measured via either a rule-based heuristic (RA-R), based on whether the answer is included in the context, or an LLM-as-Judge evaluation (RA-L), where an external large LLM assesses whether crucial evidence for answering the question is present. We use GPT-4.1 (OpenAI, 2025) as the Judge model with the prompt in Figure A5. Given RA-R and RA-L, we implement two variants of CUE: CUE-R and CUE-L.

Table 2 shows that Verbal Annotations substantially increase both CUE-R and CUE-L. We note that Verbal Annotations improve CUE-R and CUE-L more than they do answering (Exact Match & F1) and retrieval accuracies (RA). This particular increase in CUE from Verbal Annotations confirms that the structured cognitive scaffold provided by them (e.g., an explicit mention of context relevance) effectively guides the reasoning process of LLMs.

4 Methodology

Our results from the previous section strongly motivate the need for an external *Verbal Reranker*

that not only evaluates the relevance of each retrieved context but also refines it into Verbal Annotations. The mechanism of Verbal Annotations echoes the logic of neural rerankers, which evaluate query-document compatibility (Khattab and Zaharia, 2020). We now present Verbal-R3, a two-part RAG framework that consists of a Generator, an LLM that performs iterative retrieval and reasoning, and a compact Verbal Reranker.

First, we train a lightweight Verbal Reranker by distilling query-document-Verbal Annotation triplets obtained from a larger, more performant reasoning model. Then, we align the Generator LLM to maximally utilize Verbal Annotations from the Verbal Reranker, and finally, we leverage a relevance-guided test-time scaling approach that strategically allocates computational resources to achieve performance gains and efficiency.

4.1 Training Lightweight Verbal Reranker

Calling proprietary models or large open-weight models as a Verbal Reranker every time a Verbal Annotation is necessary requires an enormous amount of compute, rendering them impractical for deployment. This critical limitation accentuates the need for a lightweight alternative to enable efficient decision-making within the iterative retrieval loop. We utilize GPT-OSS-120B as the teacher model to synthesize Verbal Annotations, which are then distilled into lightweight student models with 1.5B or 3B trainable parameters suitable for high-throughput inference.

The quality of distilled annotations is highly sensitive to the teacher model’s instruction design. An extensive prompt search is performed to identify the optimal prompting strategy, evaluating candidate prompts based on their ability to elicit discriminative and critical feedback rather than generic summaries. The best-performing prompt in Figure A4 explicitly instructs the teacher model to assign a relevance score and fill logical gaps in retrieved documents.

Using this optimized configuration, we generate a large-scale corpus of synthetic query-document-Verbal Annotation triplets. Each Verbal Annotation includes both a textual critique and a scalar relevance score on a 1–5 scale, which is used for reranking documents. The quality of the resulting data is further validated by Gemini 2.5 Pro (Comanici et al., 2025), a near-human-level proprietary LLM from a different model family, tasked with assessing the reasonability of each generated

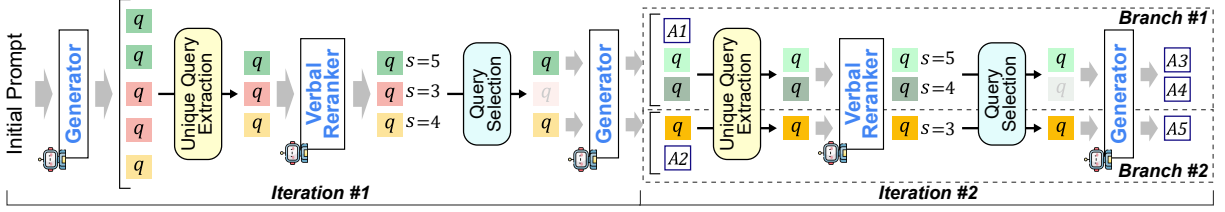


Figure 2: An example of relevance-guided test-time scaling with two iterations and two branches. Different background colors for queries (q) represent distinct queries and $A1, \dots, A5$ denote the generated answers.

Verbal Annotation, yielding a 98.5% approval rate.

A Verbal Reranker is trained with a supervised fine-tuning (SFT) objective over synthetic Verbal Annotations from the teacher model. The input is $x^{\text{VR}} = (i_{\text{VR}}, q, d)$, where i_{VR}, q, d , denote the assessing prompt, a query, and a document, respectively. The output is $y^{\text{VR}} = (v, s)$, where v and s denote a Verbal Annotation and a relevance score ($s \in \{1, 2, 3, 4, 5\}$). Given a dataset $\mathcal{D} = \{(x^{\text{VR}}, y^{\text{VR}})\}$, the training objective is expressed as:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(x^{\text{VR}}, y^{\text{VR}}) \sim \mathcal{D}} \left[\sum_{t=1}^T \log p_{\theta}(y_t^{\text{VR}} | x^{\text{VR}}, y_{<t}^{\text{VR}}) \right],$$

where p_{θ} denotes the student Verbal Reranker parameterized by θ , and T is the output sequence length. This results in a compact module capable of emulating the nuanced judgment of a 120B model while maintaining the low latency required for iterative retrieval.

4.2 Aligning Generator with Verbal Reranker via Reinforcement Learning

Preliminary. Following the training of the Verbal Reranker, we optimize the Generator to maximally utilize Verbal Annotations from the Verbal Reranker. Following the training protocol in Search-R1 (Jin et al., 2025b), the Generator policy π_{ϕ} is trained via Group Relative Policy Optimization GRPO (Shao et al., 2024) to maximize the objective below:

$$\mathcal{J}_{\text{GRPO}}(\phi) = \mathbb{E}_{\substack{x^{\text{Gen}} \sim \mathcal{D} \\ \{y_i^{\text{Gen}}\} \sim \pi_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \frac{1}{\sum_{t=1}^{|y_i^{\text{Gen}}|} I(y_{i,t}^{\text{Gen}})} \sum_{t: I(y_{i,t}^{\text{Gen}})=1}^{|y_i^{\text{Gen}}|} \min \left(r_{i,t} \hat{A}_i, \text{clip}(r_{i,t}, 1 - \epsilon, 1 + \epsilon) \hat{A}_i \right) - \beta \mathbb{D}_{\text{KL}}[\pi_{\phi} || \pi_{\text{ref}}] \right],$$

where $I(y_{i,t}^{\text{Gen}})$ is the indicator function that yields 1 only for tokens generated by the policy model

being trained and 0 otherwise (e.g., reranker responses). $r_{i,t}$ denotes the importance ratio, defined as $\frac{\pi_{\phi}(y_{i,t}^{\text{Gen}} | x^{\text{Gen}}, y_{i,<t}^{\text{Gen}})}{\pi_{\text{old}}(y_{i,t}^{\text{Gen}} | x^{\text{Gen}}, y_{i,<t}^{\text{Gen}})}$, and \hat{A}_i represents the group-relative advantage, which is computed by normalizing the sequence-level reward R_i within each group: $\hat{A}_i = \frac{R_i - \text{mean}(\{R_k\}_{k=1}^G)}{\text{std}(\{R_k\}_{k=1}^G)}$.

Trajectory Construction. The output sequence y^{Gen} is constructed iteratively through multi-turn interactions between the Generator π_{θ} and the Verbal Reranker. When the Generator produces a search query q after reasoning about the user’s question, a retriever first fetches a set of n documents $\mathcal{D} = \{d_1, d_2, \dots, d_n\}$. For each document d_i , the Verbal Reranker evaluates its relevance to q and produces a Verbal Annotation v_i and an integer relevance score $s_i \in \{1, 2, 3, 4, 5\}$:

$$(v_i, s_i) \sim p_{\theta}(\cdot | i_{\text{VR}}, q, d_i).$$

To further distinguish between documents with identical scores, we extract the logit value l_i corresponding to the predicted score token s_i from the Verbal Reranker’s output distribution.

From the candidate pool, we select the top- k Verbal Reranker responses based on the assigned scores s_i . In the event of a tie, the response with the higher generation logit l_i is prioritized. The selected responses are then appended to the sequence y in the format: "<information>... [Doc i] v_i (Relevance score: s_i)... </information>". This iterative process of reasoning, query generation, retrieval, and reranked context augmentation continues until a final answer is successfully parsed from the Generator’s output.

Reward Design. To simultaneously incentivize factual accuracy and structural consistency, we define a hierarchical reward function similar to (Jin et al., 2025a). The reward comprises the outcome reward (R_{outcome}), the format reward (R_{format}), and the answer reward (R_{answer}). The total reward R is assigned based on the correctness of the final

Method	2Wiki		Bamboogle		HotpotQA		MuSiQue		NQ		PopQA		TriviaQA		Average	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
E5 (Baseline)	13.54	18.83	08.90	16.93	23.55	32.75	04.10	08.27	33.49	44.51	37.04	43.98	53.16	62.50	24.83	32.54
MonoT5 3B	17.13	22.57	12.65	18.39	28.72	39.38	06.87	12.99	33.57	44.62	37.95	44.90	56.94	66.78	27.69	35.66
RankLLama 7B	17.54	23.39	12.60	19.32	28.44	39.05	06.74	12.27	34.60	46.50	39.55	46.71	57.69	67.56	28.17	36.40
Rank1 7B	16.86	22.16	15.75	23.60	27.70	38.07	06.95	12.68	35.82	47.65	39.71	46.73	59.38	69.46	28.88	37.19
Ours 1.5B	20.20	24.66	15.25	22.86	24.13	33.93	05.75	11.42	33.91	46.02	39.38	46.84	58.00	67.45	28.09	36.17
Ours 3B	21.74	26.31	20.35	29.64	26.47	36.99	06.91	12.97	34.90	47.51	40.77	48.41	60.16	70.02	30.19	38.84

Table 3: Evaluation results of single-turn retrieval systems augmented with various reranker models on Question-Answering benchmarks. Qwen-2.5-3B-Instruct is used as the model for answer generation.

answer and its adherence to the prescribed format:

$$R = \begin{cases} R_{\text{outcome}} & \text{Correct Answer \& Correct Format} \\ R_{\text{outcome}} - R_{\text{format}} & \text{Correct Answer \& Incorrect Format} \\ R_{\text{format}} & \text{Incorrect Answer \& Correct Format} \\ R_{\text{answer}} & \text{Incorrect Format \& Answer Parsed} \\ 0.0 & \text{Otherwise} \end{cases}$$

Detailed parsing methods and formats are provided in Section A2.

4.3 Relevance-Guided Test-Time Scaling

We propose to leverage the relevance score assigned by the Verbal Reranker as a proxy for evaluating the quality of queries produced by the Generator at inference time. This approach is grounded in the premise that a high relevance score validates the Generator’s information-seeking strategy by indicating that the query successfully guided the retriever to pertinent documents. Based on this intuition, we introduce an efficient relevance-guided test-time scaling method that selectively explores effective search trajectories.

We define each iteration t as a set of branches, where a single iteration encompasses a complete cycle of reasoning, query generation, retrieval, and reranking. Within this framework, each branch represents an independent reasoning trajectory that maintains its own accumulated historical context and serves as the fundamental unit for the query selection and path expansion processes.

Scoring and Selection. After the generation, parsed answers are added into the answer candidate set \mathcal{F} , while generated queries are utilized for the selection process described below. To selectively explore promising Generator trajectories, our method prioritizes the extraction of effective search directions, unlike naïve majority voting (Wang et al., 2023) which treats all trajectories equally.

To optimize computational efficiency, retrieval and reranking are performed only for the unique query set. For each unique query, we obtain a set of relevance scores $\mathcal{S}_q = \{s_1, s_2, \dots, s_n\}$ and the

logit l_q of $s_{\max,q}$ via the Verbal Reranker, defining $s_{\max,q} = \max(\mathcal{S}_q)$.

In the sampling step, we consider the unique query set for each branch. A query q and corresponding Verbal Annotation is selected for the next iteration with a probability $P(q)$ that reflects its relative strength within the branch:

$$P(q) = \left(\frac{s_{\max,q}}{s_{\text{best}}} \right)^\alpha,$$

where s_{best} is the highest score achieved within a branch, and α is a scaling hyperparameter. If a query is selected, a new branch for the subsequent iteration is initialized with an augmented context, constructed by appending the reasoning, the selected query, and its corresponding Verbal Annotation to the historical context of the current branch. This ensures that generation contexts are predominantly constructed along promising reasoning paths without sacrificing diversity across reasoning trajectories.

Following query selection, the total number of generations across all active branches is regulated to match the remaining budget, $N - |\mathcal{F}|$, where N is a total trajectory budget. We distribute these sample slots to ensure each branch receives as uniform a base allocation as possible. Any remaining slots from the division are sequentially assigned to branches in descending order of their relevance scores $s_{\max,q}$ and logits l_q , both of which were previously stored during the reranking.

Consolidation. This process iterates until $|\mathcal{F}| = N$ or maximum iteration T is reached. Finally, majority voting is performed over the answers collected in \mathcal{F} to determine the final output A^* . We provide the detailed workflow of relevance-guided test-time scaling in Algorithm A1, as well as an illustrative example in Figure 2.

Method	2wiki		Bamboogle		HotpotQA		MuSiQue		NQ		PopQA		TriviaQA		Average	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Qwen2.5-3B																
Direct Answer	18.47	22.89	3.20	7.87	12.23	19.08	1.57	6.78	9.14	15.21	7.02	11.01	22.73	30.51	10.62	16.19
RAG	13.54	18.83	8.90	16.93	23.55	32.75	4.10	8.27	33.49	44.51	37.04	43.98	53.16	62.50	24.83	32.54
IRCoT	17.37	24.89	16.90	25.10	21.82	30.60	5.09	9.95	23.13	33.39	32.23	39.33	40.18	49.77	22.26	30.49
ITER-RETGEN	23.83	29.41	12.55	21.51	28.01	37.21	5.96	11.21	36.81	46.55	41.61	47.02	55.86	64.80	29.23	36.82
Search-R1	38.06	44.10	32.75	42.73	37.50	47.83	14.69	22.36	43.49	51.68	43.98	47.94	60.75	67.52	38.75	46.31
Verbal-R3 (Ours)	43.84	51.00	49.25	61.34	44.47	56.95	20.89	29.99	46.01	56.31	47.59	52.85	65.50	74.15	45.36	54.66
Qwen2.5-7B																
Direct Answer	23.12	28.47	10.25	16.89	17.14	25.07	3.60	11.69	12.58	21.63	14.54	18.89	38.02	45.35	17.04	24.00
RAG	20.46	25.76	14.15	23.29	27.64	36.62	5.13	9.39	31.47	42.31	37.79	45.16	55.75	64.52	27.48	35.92
IRCoT	14.02	21.84	18.30	28.67	23.89	33.98	5.05	10.22	23.71	37.26	32.51	40.62	49.85	60.75	23.90	33.33
ITER-RETGEN	24.99	31.74	20.65	30.75	31.72	42.15	6.99	14.37	36.37	47.41	40.91	47.80	60.02	69.10	31.66	40.47
Search-R1	40.17	46.55	40.90	50.95	41.22	52.77	18.78	27.94	44.93	53.51	44.23	48.30	63.10	70.71	41.90	50.10
Verbal-R3 (Ours)	47.19	53.89	53.55	65.25	47.47	60.32	25.16	34.85	48.28	57.87	48.30	52.77	68.13	76.03	48.30	57.28

Table 4: Evaluation of RAG systems on single- and multi-hop Question-Answering benchmarks. Comparison with recent agentic methods is included in A7.

5 Experimental Results and Discussion

5.1 Experimental Details

Benchmarks. We primarily evaluate models across seven QA benchmarks, categorized into single-hop tasks, namely NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2023); and multi-hop reasoning tasks, comprising 2WikiMultiHopQA (Ho et al., 2020), Bamboogle (Press et al., 2023a), MuSiQue (Trivedi et al., 2022), and HotpotQA (Yang et al., 2018). For all datasets, performance is measured using Exact Match (EM) and F1 score. Since the Bamboogle dataset consists of only 125 samples, we report the average performance over 16 sampling trials to ensure the statistical robustness and reliability of our evaluation results. To separately assess the reranking ability, the BEIR benchmark (Thakur et al., 2021) is used, with nDCG@10 as the main evaluation metric for reranking performance.

Baselines. To evaluate the performance of our Verbal Reranker, we compare it against existing reranker models: MonoT5 (Nogueira et al., 2019), RankLLaMA (Ma et al., 2024), and Rank1 (Weller et al., 2025). To evaluate the effectiveness of Verbal-R3 as a standalone RAG system, we compare it against representative baselines in RAG: Direct Answer, Standard RAG, IRCoT (Trivedi et al., 2023), ITER-RETGEN (Shao et al., 2023), and Search-R1 (Jin et al., 2025b). Implementation details for these baselines are provided in Section A1.1. Finally, our test-time scaling approach is compared against two variations of naïve Majority Voting, whose descriptions are also given in Section A1.1.

Training & Inference Settings. Following Jin et al. (2025b), we use E5 (Wang et al., 2022) as our

base retriever across all experiments.

- **Verbal Reranker Training:** We distill the outputs of GPT-OSS-120B on the Natural Question(NQ) training set to Qwen2.5-Instruct Model (1.5B & 3B) (Qwen et al., 2025).

- **Generator Training:** As the Generator of Verbal-R3, we employ Qwen2.5-3B and Qwen2.5-7B (Qwen et al., 2025) as backbones, using NQ and HotpotQA as the training datasets.

- **Verbal-R3 Inference:** The 3B Verbal Reranker is used as the default Verbal Reranker for Verbal-R3, and $n = 15$ and $k = 3$ are used. Thus, Verbal-R3 3B and 7B denote that 3B and 7B Generators are used, respectively, with the size of the Verbal Reranker fixed as 3B. For test-time scaling methodology, we configure the scaling hyperparameter α to 7.5 and set the total trajectory budget N to 5.

More details on training and inference are provided in Sections A1.2 and A1.3.

5.2 Results

Reranker. We separately assess the effectiveness of our Verbal Reranker by combining it with single-turn RAG frameworks in a plug-and-play manner. The results, reported in Table 3 in terms of EM and F1 scores on QA benchmarks, indicate that leveraging our 1.5B-parameter Verbal Reranker achieves performance comparable to larger 3B- and 7B-parameter models, and using our 3B-parameter Verbal Reranker consistently outperforms all baselines. We additionally compare the stand-alone reranking performance of our Verbal Reranker on the BEIR benchmark. The results, along with the discussion, are provided in Section A3.

Verbal-R3. Table 4 presents EM and F1 scores of the Verbal-R3 framework and baselines on seven QA benchmarks. Verbal-R3 robustly achieves

Method	2wiki		Bamboogle		HotpotQA		MuSiQue		NQ		PopQA		TriviaQA		Average		
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	# Calls
Verbal-R3 3B																	
No M.V.	43.84	51.00	49.25	61.34	44.47	56.95	20.89	29.99	46.01	56.31	47.59	52.85	65.50	74.15	45.36	54.66	2.27
Naïve M.V.																	
w/o U.Q.E.	46.18	53.10	51.45	62.69	46.52	58.96	23.46	32.64	47.76	57.76	48.62	53.89	66.83	75.24	47.26	56.33	11.28
w/ U.Q.E.	45.17	52.15	50.40	62.48	46.06	58.45	23.42	32.32	47.06	57.33	48.54	53.69	66.20	74.74	46.69	55.88	6.52
Relevance-Guide (Ours)	45.53	52.57	50.60	63.01	46.58	59.29	24.99	33.99	47.95	57.76	48.22	53.54	66.42	74.99	47.18	56.45	6.18
Verbal-R3 7B																	
No M.V.	47.19	53.89	53.55	65.25	47.47	60.32	25.16	34.85	48.28	57.87	48.30	52.77	68.13	76.03	48.30	57.28	1.82
Naïve M.V.																	
w/o U.Q.E.	49.67	56.35	53.25	65.75	50.10	62.74	27.35	37.41	50.00	59.52	49.90	54.22	69.25	77.14	49.93	59.02	9.11
w/ U.Q.E.	48.95	55.71	52.45	65.20	49.44	61.91	26.77	36.79	49.31	58.91	49.11	53.53	68.60	76.42	49.23	58.35	4.63
Relevance-Guide (Ours)	49.90	56.66	53.45	65.80	50.32	63.22	28.96	38.69	49.25	58.93	49.71	54.20	69.27	77.24	50.12	59.25	4.21

Table 5: Comparison of different test-time scaling approaches. $N = 5$ is used for our relevance-guided approach. M.V. denotes majority voting and U.Q.E. denotes unique query extraction.

state-of-the-art performance across all evaluated benchmarks, outperforming all baselines, including Search-R1, the most competitive of them. Notably, Verbal-R3 3B, which utilizes a total of $\sim 6B$ parameters (3B Generator + 3B Verbal Reranker) not only yields a 17.1% increase in EM and an 18.0% improvement in F1 over Search-R1 3B but also surpasses the Search-R1 7B by 8.26% in EM and 9.10% in F1. As for Verbal-R3 7B, it improves EM and F1 of Search-R1 7B by 15.3% and 14.3%, respectively.

We also highlight that Verbal-R3 exhibits noticeably larger improvements on multi-hop benchmarks. Verbal-R3 3B model improves the average F1 score on multi-hop tasks by 26.91%, which is nearly threefold the 9.67% gain observed in single-hop tasks. This trend remains consistent for the 7B model, with a 20.26% improvement for multi-hop versus 8.20% for single-hop. These results indicate that mitigating the noise and redundancy in raw retrieval observations with our Verbal Reranker is particularly crucial in multi-hop scenarios, where the Generator’s context becomes increasingly saturated with a vast amount of external data accumulated through successive search iterations.

Experiments on a train-free variant of Verbal-R3 with a larger backbone model in A6 further demonstrate the scalability and cost-efficiency of our method.

Relevance-Guided Test-time Scaling. Table 5 shows the results of using our test-time scaling method. When compared to the naïve majority voting without unique query extraction, our relevance-guided method achieves comparable or superior performance while reducing the number of reranker calls by 45.2% and 53.8% for the 3B and 7B models, respectively. Furthermore, our approach outperforms the naïve majority voting with unique

query extraction, demonstrating a robust reduction in reranker calls alongside improved accuracy.

To quantify the computational efficiency, we empirically measure the average number of generated tokens by the reranker, amounting to approximately 990 tokens per call when $n = 15$. By adopting our prioritized selection strategy, we save approximately 376 and 4,950 tokens per query compared to the naïve baselines with and without unique query extraction, respectively. These results collectively demonstrate that the relevance-guided test-time scaling effectively prioritizes promising queries and eliminates unnecessary operations, thereby ensuring a highly efficient inference process without compromising, and in many cases enhancing, the overall RAG performance.

5.3 Analysis

Ablation Study on Verbal Annotations. To investigate the impact of Verbal Annotation, we compared our proposed method against a variant trained with full document observations instead of Verbal Annotations, while keeping the same reranking strategy for both. As shown in Table A2, the model trained with Verbal Annotations consistently and robustly outperforms the full-document counterpart across all evaluated benchmarks. This performance gap underscores the critical role of Verbal Annotation in the RAG pipeline and validates that the inclusion of verbal feedback is essential for bridging the gap between retrieval and reasoning.

Verbal-R3’s Context Utilization. We adopt analytical metrics in Section 3 to study Verbal-R3. Table 6 presents a comparative analysis of Search-R1 3B, Verbal-R3 3B, and an ablated version of Verbal-R3 3B that operates on the full document. The results show that both Verbal-R3 and its ablation achieve higher retrieval accuracy than Search-

Model	EM	F1	RA-R	RA-L	CUE-R	CUE-L
Search-R1	38.74	46.31	61.79	48.26	64.77	69.04
Verbal-R3 <i>w/o</i> VA	42.73	51.75	66.73	53.52	65.21	69.76
Verbal-R3	45.36	54.66	69.06	52.86	67.54	71.64

Table 6: Comparison of Verbal-R3, its ablation variant without Verbal Annotation, and Search-R1.

R1, indicating that the performance gains primarily stem from the reranking component of the Verbal Reranker. Verbal-R3 consistently attains higher CUE scores independent of retrieval accuracy, suggesting that the performance gains largely stem from improved context utilization.

6 Conclusion

This work unveiled the importance of Verbal Annotations, analytic narratives that serve as the logical bridge between a search query and retrieved contexts, in improving RAG. Based on this finding, we proposed Verbal-R3, an agentic RAG framework consisting of a Generator that performs iterative retrieval and reasoning and a Verbal Reranker that produces Verbal Annotations to guide the Generator. Verbal-R3 is complemented by novel relevance-guided test-time scaling that efficiently allocates test-time computation to promising reasoning trajectories. The success of Verbal-R3 at attaining state-of-the-art performance on complex QA benchmarks, along with thorough ablative studies, validate the contribution of each proposed technical component. In summary, Verbal-R3 is an important step towards developing an interpretable and controllable agentic RAG framework that closes the gap between retrieval and LLM reasoning.

Limitations & Potential Risks

Our approach employs two LLM modules for RAG. While we develop a lightweight reranker by distilling the outputs of a larger reasoning model, utilizing an additional LLM introduces additional computational overhead depending on the deployment setting. Although our Verbal Reranker is quite performant, it is still far from being perfect. Therefore, the remaining errors present in the Verbal Annotations will accumulate during iterative retrieval and reasoning. Due to practical constraints, our experiments were conducted using an offline retriever and publicly available datasets. Exploring the integration of a live web-search-based retrieval system is left for future study.

Though RAG reduces hallucination, the inherent probabilistic nature of LLMs inhibits a complete

elimination of hallucinations. In particular, a user can still receive inaccurate, misleading, or fabricated information when retrieved documents are incomplete, outdated, irrelevant, or incorrectly interpreted by the model. Additionally, errors may arise from improper retrieval ranking, ambiguous user queries, or overconfidence in generated responses. These risks highlight the necessity of complementary safeguards, such as confidence scoring, human oversight, and continuous evaluation of retrieval quality, especially in high-stakes domains. Addressing these limitations is essential to ensure the reliability of RAG-based systems.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University); No.2022-0-00959, RS-2022-II220959], by the National Research Foundation of Korea (NRF) grant [No.2022R1A3B1077720, 2022R1A5A7083908], BK21 FOUR Program of the Education and Research Program for Future ICT Pioneers, Seoul National University in 2026, and by the Samsung Electronics Co., Ltd [IO250520-12926-01].

References

- Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Cheung. 2022. [Why exposure bias matters: An imitation learning perspective of error accumulation in language generation](#). In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 700–710, Dublin, Ireland. Association for Computational Linguistics.
- Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. [Self-RAG: Learning to retrieve, generate, and critique through self-reflection](#). In *The Twelfth International Conference on Learning Representations*.
- Gheorghe Comanici, Eric Bieber, Mike Schaeckermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3416 others. 2025. [Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities](#). *Preprint*, arXiv:2507.06261.

- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023a. [Precise zero-shot dense retrieval without relevance labels](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.
- Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1).
- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. [Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps](#). In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.
- OpenAI Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mkadry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alexander Kirillov, Alex Nichol, Alex Paino, and 397 others. 2024. Gpt-4o system card. *ArXiv*, abs/2410.21276.
- Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.
- 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.
- Bowen Jin, Jinsung Yoon, Priyanka Kargupta, Sercan O Arik, and Jiawei Han. 2025a. An empirical study on reinforcement learning for reasoning-search interleaved llm agents. *arXiv preprint arXiv:2505.15117*.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025b. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.
- 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.
- Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. [Efficient memory management for large language model serving with pagedattention](#). In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP '23*, page 611–626, New York, NY, USA. Association for Computing Machinery.
- 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.
- Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang, Qianben Chen, Weichen Sun, Qiexiang Wang, Hongxuan Lu, Tianrui Qin, Chenghao Zhu, Yi Yao, Shuying Fan, Xiaowan Li, Tiannan Wang, Pai Liu, King Zhu, and 11 others. 2025a. [Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic rl](#). *Preprint*, arXiv:2508.13167.
- Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025b. [Search-ol: Agentic search-enhanced large reasoning models](#). *CoRR*, abs/2501.05366.

- Xuanzhang Liu, Jianglun Feng, Zhuoran Zhuang, Junzhe Zhao, Maofei Que, Jieting Li, Dianlei Wang, Hao Tong, Ye Chen, and Pan Li. 2026. [Coda: A context-decoupled hierarchical agent with reinforcement learning](#). In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining, WSDM '26*, page 436–446, New York, NY, USA. Association for Computing Machinery.
- Xing Han Lù. 2024. Bm25s: Orders of magnitude faster lexical search via eager sparse scoring. *arXiv preprint arXiv:2407.03618*.
- Kun Luo, Hongjin Qian, Zheng Liu, Ziyi Xia, Shitao Xiao, Siqi Bao, Jun Zhao, and Kang Liu. 2025. [Inflow: Reinforcing search agent via reward density optimization](#). *Preprint*, arXiv:2510.26575.
- Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023a. [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.
- Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425.
- Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023b. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156*.
- 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.
- Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.
- Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. [Document ranking with a pre-trained sequence-to-sequence model](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718, Online. Association for Computational Linguistics.
- Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.
- OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, and 108 others. 2025. [gpt-oss-120b gpt-oss-20b model card](#). *Preprint*, arXiv:2508.10925.
- OpenAI. 2025. Introducing gpt-4.1 in the api. <https://openai.com/index/gpt-4-1/>.
- Ronak Pradeep, Sahel Sharifmoghammad, and Jimmy Lin. 2023a. Rankvicuna: Zero-shot listwise document reranking with open-source large language models. *arXiv preprint arXiv:2309.15088*.
- Ronak Pradeep, Sahel Sharifmoghammad, and Jimmy Lin. 2023b. Rankzephyr: Effective and robust zero-shot listwise reranking is a breeze! *arXiv preprint arXiv:2312.02724*.
- Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023a. [Measuring and narrowing the compositionality gap in language models](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.
- Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2023b. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711.
- 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, Jiayi Yang, Jingren Zhou, and 25 others. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.
- Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.
- Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. [Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274, Singapore. Association for Computational Linguistics.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. 2024. [Deepseekmath: Pushing the limits of mathematical reasoning in open language models](#).
- Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. [Is ChatGPT good at search?](#)

- investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore. Association for Computational Linguistics.
- Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. **MuSiQue: Multi-hop questions via single-hop question composition**. *Transactions of the Association for Computational Linguistics*, 10:539–554.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. **Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions**. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. **Self-consistency improves chain of thought reasoning in language models**. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.
- Orion Weller, Kathryn Ricci, Eugene Yang, Andrew Yates, Dawn Lawrie, and Benjamin Van Durme. 2025. Rank1: Test-time compute for reranking in information retrieval. *arXiv preprint arXiv:2502.18418*.
- Jun Xu, Xinkai Du, Yu Ao, Peilong Zhao, Yang Li, Ling Zhong, Lin Yuan, Zhongpu Bo, Xiaorui Wang, Mengshu Sun, Zhengke Gui, Dalong Zhang, Zhaoyang Wang, Wang Qiwei, Yangyang Hou, Zhiying Yin, Haofen Wang, Huajun Chen, Lei Liang, and Jun Zhou. 2026. **Thinker: Training llms in hierarchical thinking for deep search via multi-turn interaction**. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40:34142–34150.
- 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, and 41 others. 2025. Qwen3 technical report. *ArXiv*, abs/2505.09388.
- 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.
- Soyoung Yoon, Eunbi Choi, Jiyeon Kim, Hyeonung Yun, Yireun Kim, and Seung-won Hwang. 2024. List5: Listwise reranking with fusion-in-decoder improves zero-shot retrieval. *arXiv preprint arXiv:2402.15838*.
- Tian Yu, Shaolei Zhang, and Yang Feng. 2024. **Autrag: Autonomous retrieval-augmented generation for large language models**.
- Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.
- Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. **DeepResearcher: Scaling deep research via reinforcement learning in real-world environments**. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 414–431, Suzhou, China. Association for Computational Linguistics.
- Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. 2023. Rankt5: Fine-tuning t5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313.

Appendix

A1 Experimental Details

A1.1 Baselines

The baselines for rerankers are as follows:

- MonoT5 (Nogueira et al., 2019) uses a sequence-to-sequence model to evaluate a document–query pair. It is prompted with the query and the document, and then computes the logit of the output sequence.
- RankLLaMA (Ma et al., 2024) leverages a large language model to build a linear regressor.
- Rank1 (Weller et al., 2025) uses test-time scaling with a large language model to produce more precise reranking.

The baselines for evaluation on QA benchmarks are as follows:

- Direct Answer assesses the generator’s internal knowledge without using any external context.
- Naïve RAG is a foundational baseline that appends retrieved documents in a single-step process.
- IRCOT (Trivedi et al., 2023) uses a training-free iterative RAG framework.
- ITER-RETGEN (Shao et al., 2023) uses an iterative approach where the model’s previous generation is utilized as context for retrieval.
- Search-R1 is a competitive baseline designed specifically for iterative RAG, optimizing multi-step search and reasoning via reinforcement learning.

The naïve majority voting baselines for test-time scaling method are as follows:

- w/ unique query extraction identifies identical queries within N sampling trials and invokes the reranker only once per unique query. The resulting response is then replicated across all redundant instances.
- w/o unique query extraction executes a reranker call for every generated query, regardless of duplication.

While w/o unique query extraction approach incurs a higher number of reranker calls, it potentially increases response diversity by leveraging the stochasticity inherent in the Verbal Reranker’s own sampling process.

A1.2 Training Details

Reranker Training. To train the Verbal Reranker, we utilize responses generated by GPT-OSS-120B on the Natural Questions (NQ) training set, which were subsequently filtered as described in the previous section. Specifically, we sampled 10,000 questions from NQ and retrieved the top 50 documents for each question using the E5 retriever, resulting in 500,000 question-document pairs. These pairs were then input as prompts to GPT-OSS-120B to generate Verbal Annotations. After applying our filtering procedure and splitting, 380,460 pairs (77%) were selected for training. We trained Qwen2.5-Instruct-1.5B and Qwen2.5-Instruct-3B on this filtered dataset for 3 epochs using a batch size of 128.

Verbal-R3 Training. The training dataset is curated from the training sets of NQ and HotpotQA, and the Generators were trained for 2 epochs with a batch size of 128 and a temperature of 1.0. To ensure the model focuses on challenging reasoning paths, we evaluated each training sample from NQ and HotpotQA using the base Qwen2.5-3B model across 10 independent sampling trials; samples that were correctly answered in at least one trial were considered "trivial" and subsequently filtered out. However, we exempted binary questions from this filtering to prevent the false exclusion of valid reasoning instances due to high random guessing probabilities.

For the Group Relative Policy Optimization (GRPO) objective, we configured the hyperparameters with $G = 5$, $\epsilon = 0.2$, and $\beta = 0.001$. Regarding the reward components, we assigned value of $R_{\text{outcome}} = 1.0$, $R_{\text{format}} = 0.2$, and $R_{\text{answer}} = 0.1$. We employ 1.5B Verbal Reranker during the training phase to minimize computational overhead. During training, we set the number of retrieved documents to $n = 15$, from which the top $k = 3$ reranker responses were integrated into the generator’s context.

A1.3 Inference Details

For all Verbal-R3 inference experiments, we employ sampling with a temperature of 1.0 and a top- p

Model	ArguA	ClimF	DBP	FiQA	NFCorp	SciDoc	SciFact	Touche	TrecC	Avg
BM25S	47.2	18.6	32.0	25.4	34.3	16.5	69.1	34.7	68.8	38.5
MonoT5-3B	42.5	25.4	44.5	46.5	37.8	19.3	76.1	30.7	79.6	44.7
RankLLaMA-7B	54.4	23.2	43.7	42.1	27.0	16.6	71.1	41.4	80.2	44.4
RankLLaMA-13B	49.3	24.5	44.9	44.1	28.1	18.1	72.7	39.2	80.8	44.6
Rank1-7B	42.8	15.0	38.9	39.5	36.2	17.2	77.2	22.8	81.9	41.3
Rank1-14B	45.3	16.2	37.4	37.9	35.8	17.9	77.0	27.1	78.2	41.4
Rank1-32B	57.6	15.8	40.7	41.8	36.9	19.6	76.8	19.9	81.9	43.4
Ours-1.5B	24.0	13.7	32.8	30.1	33.5	15.1	69.4	23.5	75.7	35.3
Ours-3B	26.3	14.5	34.4	35.0	34.0	14.6	70.8	22.3	76.9	36.5

Table A1: The nDCG@10 performance of Verbal Reranker in BEIR benchmark. Baseline results originate from Weller et al. (2025).

Method	2wiki		Bamboogle		HotpotQA		MuSiQue		NQ		PopQA		TriviaQA		Average	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Verbal-R3 3B	43.84	51.00	49.25	61.34	44.47	56.95	20.89	29.99	46.01	56.31	47.59	52.85	65.50	74.15	45.36	54.66
w/o Verbal Annotation	39.09	45.02	43.55	56.57	43.74	55.79	20.07	28.37	45.12	54.97	44.96	50.17	62.61	71.39	42.73	51.75

Table A2: Ablation study using Verbal-R3 3B.

of 0.95. The 3B Verbal Reranker was tasked with evaluating a candidate pool of $n = 15$ retrieved documents. From this pool, the top $k = 3$ reranked results were selected to be appended to the generation context for the Verbal-R3 Generator. To generate outputs from the Verbal Reranker, we used a temperature of 0.6 and top- $p = 0.95$. Inference was performed via the vLLM (Kwon et al., 2023) for optimized computational performance.

A2 Parsing Methods and Formats

Following the protocol of Search-R1, we parse search queries only when they are enclosed within `<search>` and `</search>` tags. Similarly, the final answer is extracted from the content within `<answer>` and `</answer>` tags. A correctly formatted response is defined as a sequence of repeated reasoning segments and `<search>` {query} `</search>` blocks, concluding with a final `<answer>` {answer} `</answer>` tag.

A3 Evaluation on Reranking Benchmarks

Table A1 reports the performance of the Verbal Reranker on the BEIR benchmark. Following (Weller et al., 2025), all models rerank the top 100 documents initially retrieved by BM25S (Lù, 2024). The results indicate that performance trends on BEIR do not align with the RAG results shown in Table 3. This discrepancy suggests that existing reranking benchmarks may be inadequate for evaluating high-quality reranking modules, consistent with the observations of Weller et al. (2025).

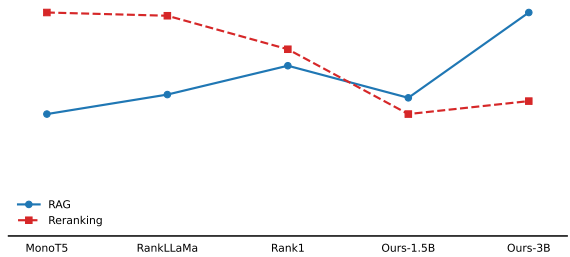


Figure A1: Performance of various rerankers on reranking benchmarks and in RAG settings. The divergent trends indicate that reranking benchmark scores do not reliably predict end-to-end RAG performance.

Figure A1 illustrates the rerankers’ performance on reranking benchmarks and in RAG settings. This further corroborates that BEIR benchmark scores do not reliably reflect a reranker’s ultimate contribution to end-to-end RAG performance.

A4 Relevance-Guided Test-Time Scaling Algorithm

The detailed algorithm for Relevance-Guided Test-Time Scaling is presented in Algorithm A1.

A5 Ablation Study on Verbal Annotations

The results of ablation study on Verbal Annotations are shown in Table A2.

Algorithm A1 Relevance-Guided Test-Time Scaling

Require: Initial context C , total trajectory budget N , max turns T , scaling hyperparameter α **Ensure:** Final answer A^*

```
1: Initialize:  $\mathcal{B}_0 \leftarrow \{(C, s = 0, l = 0)\}$ ,  $\mathcal{F} \leftarrow \emptyset$ ,  $t \leftarrow 0$ 
2: while  $t < T$  and  $|\mathcal{F}| < N$  do
3:    $N_e \leftarrow N - |\mathcal{F}|$  ▷ Calculate remaining budget
4:   Sort branches  $b_i = (C_i, s_i, l_i) \in \mathcal{B}_t$  in descending order of  $s_i$ , then  $l_i$ 
5:    $n_{\text{base}} \leftarrow \lfloor N_e / |\mathcal{B}_t| \rfloor$ ,  $\rho \leftarrow N_e \pmod{|\mathcal{B}_t|}$ 
6:   for all  $b_i \in \mathcal{B}_t$  do ▷ Step 1: Generation
7:      $m_i = n_{\text{base}} + \mathbb{1}(i < \rho)$ 
8:     Generate  $m_i$  responses  $\{y_{i,j}\}_{j=1}^{m_i}$  with context  $C_i$ 
9:      $\mathcal{M}_i \leftarrow \emptyset$  ▷ Initialize query-response mapping
10:    for all  $y_{i,j}$  do
11:      if  $y_{i,j}$  contains answer  $a_{i,j}$  then
12:         $\mathcal{F} \leftarrow \mathcal{F} \cup \{a_{i,j}\}$ 
13:      else if  $y_{i,j}$  contains query  $q_{i,j}$  then
14:        Collect  $q_{i,j}$  for reranking
15:         $\mathcal{M}_i(q_{i,j}) \leftarrow y_{i,j}$ 
16:     $\mathcal{V} \leftarrow \text{Unique}(\text{all collected } q_{i,j})$  ▷ Step 2: Reranking
17:    for all  $v \in \mathcal{V}$  do
18:       $S_v \leftarrow \text{Rerank}(v)$ ,  $s_{\text{max},v} \leftarrow \max(S_v)$ 
19:       $l_v \leftarrow \text{GetRerankerLogit}(s_{\text{max},v})$ 
20:     $\mathcal{B}_{t+1} \leftarrow \emptyset$ 
21:    for all  $b_i \in \mathcal{B}_t$  do ▷ Step 3: Branch-wise Sampling
22:       $\mathcal{U}_i \leftarrow \text{Unique}(\{q_{i,j} \in \text{responses of } b_i\})$ 
23:       $s_{\text{best},b_i} \leftarrow \max_{q \in \mathcal{U}_i} \{s_{\text{max},q}\}$ 
24:      for all  $q \in \mathcal{U}_i$  do
25:         $P(q) \leftarrow \left(\frac{s_{\text{max},q}}{s_{\text{best},b_i}}\right)^\alpha$ 
26:        Sample  $z \sim \text{Bernoulli}(P(q))$ 
27:        if  $z = 1$  then
28:           $y \leftarrow \mathcal{M}_i(q)$ ,  $C' \leftarrow C_i \oplus y \oplus \text{Obs}_q$ 
29:           $b_{\text{new}} \leftarrow (C', s_{\text{max},q}, l_q)$ 
30:           $\mathcal{B}_{t+1} \leftarrow \mathcal{B}_{t+1} \cup \{b_{\text{new}}\}$ 
31:     $t \leftarrow t + 1$ 
32: return  $A^* = \text{MajorityVote}(\mathcal{F})$ 
```

Method	2Wiki		Bamboogle		HotpotQA		MuSiQue		NQ		PopQA		TriviaQA		Average		FLOPs ($\times 10^{15}$)
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	
w/o Reranker	35.7	42.2	39.8	53.0	40.4	52.3	17.8	26.1	40.3	52.8	44.1	51.0	66.4	74.7	40.6	50.3	0.207
w/ 3B Verbal Reranker	37.3	43.6	47.4	59.5	41.1	53.2	18.3	27.0	40.3	52.6	44.2	51.6	67.2	75.6	42.3	51.9	0.236

Table A3: Evaluation in the iterative retrieval setting with Qwen2.5-32B-Instruct as the Generator, with and without our 3B Verbal Reranker.

A6 Analysis of Scaling and Computational Cost

To prioritize research efficiency, we restrict RL-based generator training to the 3B and 7B models. To further analyze generator scaling and the computational cost of the Verbal Reranker, we additionally evaluate Qwen2.5-32B-Instruct model as the Generator on the RAG benchmarks and report the corresponding inference cost in Table A3. Under the iterative retrieval setting, the results show that our Verbal Reranker consistently improves both EM and F1 even when paired with a 32B Generator, demonstrating that the benefits of our Ver-

bal Reranker transfer robustly beyond the model scales.

From the perspective of computational efficiency, incorporating the Verbal Reranker increases the average F1 score by 3.13% while introducing only 13.8% additional FLOPs. In contrast, scaling Search-R1 from a 3B to a 7B backbone yields an 8.18% F1 improvement at the cost of approximately 133% more FLOPs. This comparison highlights that our approach provides a more favorable performance-cost trade-off. Moreover, because the Verbal Reranker remains a relatively small external module, its relative computational overhead

becomes less significant as the Generator size increases.

A7 Comparisons with Advanced Agentic RAG Frameworks

In addition to Search-R1, which is used as a baseline in the main paper, we further compare our method against advanced agentic RAG frameworks. Specifically, we include AFM (Li et al., 2025a), CoDA (Liu et al., 2026), Thinker (Xu et al., 2026), and InfoFlow (Luo et al., 2025) as additional baselines. The comparison results are presented in Table A4. For these baselines, we report the performance numbers as provided in their original papers.

A8 Use of AI Writing Assistant

AI-driven writing assistant tools, such as ChatGPT, Gemini, and Claude, were used to paraphrase sentences for improved clarity and to search for synonyms or similar phrases. Beyond this, no additional AI assistance was involved during the research process or paper write-up.

Method	2Wiki	Bamboogle	HotpotQA	MuSiQue	NQ	PopQA	TriviaQA	Average
<i>Qwen2.5-3B</i>								
AFM	43.4	45.6	41.1	19.0	39.3	42.4	58.2	41.3
CoDA	40.8	36.8	40.7	18.7	44.1	45.2	60.8	41.0
Thinker	46.9	42.4	40.0	21.4	43.9	46.9	59.8	43.0
InfoFlow	45.2	41.2	44.6	21.0	44.5	47.0	63.7	43.9
Verbal-R3 (Ours)	43.8	49.3	44.5	20.9	46.0	47.6	65.5	45.4
<i>Qwen2.5-7B</i>								
AFM	49.2	49.6	43.9	22.3	43.9	46.5	63.3	45.5
Thinker	46.9	48.0	42.1	22.1	45.0	48.4	64.2	45.2
InfoFlow	47.2	47.6	44.3	21.9	47.2	48.1	68.1	46.3
Verbal-R3 (Ours)	47.2	53.6	47.5	25.2	48.3	48.3	68.1	48.3

Table A4: Comparison with agentic RAG frameworks under 3B and 7B backbone settings. AFM reports performance using an LLM-as-a-Judge metric, whereas the remaining methods are evaluated with Exact Match.

Prompt for Naive Context Integration

```
{
  "role":"system",
  "content":"You are a helpful assistant"
},
{
  "role":"user",
  "content":"Retrieved results:
Document #1. {document1}
Document #2. {document2}
Document #3. {document3}

Question: {question}

Answer the question, using information in the retrieved results. Do not answer as 'unknown' or 'i don't know'.
Give the answer that is the most probable based on the given information.
You must write the answer in between the <answer> and </answer> tags. Example: <answer>Boston</answer>

Again, do not answer as 'unknown' or 'i don't know'."
}
```

Figure A2: Prompt template for naïve context integration.

Prompt for Generating a Paraphrased Retrieved Context

```
{
  "role": "system",
  "content": "You are a helpful assistant that summarizes documents concisely."
},
{
  "role": "user",
  "content": ""Please summarize the following document, focusing on the key information.
Remove terms like 'this document':

Document:
{document}""
}
```

Figure A3: Prompt template for generating paraphrased contexts.

Prompt for Obtaining Verbal Annotations and a Relevance Score with a Verbal Reranker

```
{
  "role": "system",
  "content": ""
}
You are an evaluator that judges how informative a document is for answering a given question.
You will receive a Question and a Document.

Carefully assess relevance and usefulness with a brief, evidence-based comment that supports downstream filtering
(mention concrete entities/claims/dates/metrics when helpful), and then a score of relevance.

Scoring rubric (1-5):
1 - Unrelated: The document has nothing to do with the question.
    It does not contain any potentially relevant information or an answer to the question.
2 - Loosely related: Contains information that might potentially help or include the answer to the question,
    but is unlikely to do so.
3 - Partially informative: Contains information that can potentially help answer the question in some way.
4 - Substantively informative: Related to the question and includes information that is relevant to it.
5 - Direct answer: Clearly related and includes key information that can be used to directly answer the question.

Output format (exactly):
Comment: <concise assessing of the document, citing specific evidence from the document;
    e.g., "Since the document states A and B, it is relevant to the question about C.">
Score: <1-5>""
},
{
  "role": "user",
  "content": ""Question: {question}
Document: {document}""
}
```

Figure A4: Prompt template for generating Verbal Annotations and a relevance score with a Verbal Reranker.

Prompt for Calculating Retrieval Accuracy through LLM-as-a-Judge Evaluation

```
{
  "role": "user",
  "content": ""You are an evaluation model.
Your job is to decide whether the retrieved results provide enough evidence to support the proposed answer to
the question. The retrieved results appear inside the model's output reasoning trace.

### Inputs

* Question: The question being asked.
* Answer: The proposed answer to the question.
* Model's Reasoning Trace:

    * The model's internal reasoning is enclosed between `` and `</think>` tags.
    * Any search queries are enclosed between `` and `</search>` tags.
    * All retrieved documents are enclosed between `` and `</information>` tags.

### Evaluation Rules

1. Output yes if the retrieved results contain clear, explicit, and direct evidence supporting the answer.
2. Output no if:

    * The retrieved results do not mention the answer, or
    * The answer is mentioned but the retrieved results do not include the factual support or justification
needed to answer the question, or
    * The answer is a Yes/No or multiple-choice selection and the retrieved results do not clearly justify
that choice.
3. Do not use the reasoning trace or the answer itself to judge correctness; evaluate based only on the
retrieved results as evidence.
4. Think step-by-step about whether the key evidence is present.
5. After your reasoning, output your final decision as exactly yes or no on a new line after the
step-by-step thinking.

Question: {question}
Answer: {answer}
Model's Reasoning Trace: {response}""
}
```

Figure A5: Prompt template for evaluating retrieval accuracy through LLM-as-a-Judge evaluation.

Prompt for Generator Reasoning

You are an expert in retrieval who specializes in thinking, searching, and answering.

Before performing any search or providing an answer, you must first reason about your next action, including before your initial search.

After reasoning, if your knowledge is not enough to answer the question, you must call the reranker function by placing your query between <search> and </search>. The reranker function will return the relevant documents with their relevance scores(1~5 scale, 5 for most relevant and 1 for least relevant) between <information> and </information>.

If the retrieved results do not contain enough information for answering the question, perform additional searches to gather more context.
Continue searching iteratively until you have gathered sufficient information to respond accurately.

Once you determine that no further external information is required, provide your answer directly within <answer> and </answer>, without detailed explanations. For example:\n<answer> Beijing </answer>.

Question: {Question}

Figure A6: Prompt template for iterative retrieval and reasoning with the Generator.