DF-RAG: Query-Aware Diversity for Retrieval-Augmented Generation

Anonymous ACL submission

Abstract

Retrieval-augmented generation (RAG) is a common technique for grounding language model outputs in domain-specific information. However, RAG is often challenged by reasoning-intensive question-answering (QA), since common retrieval methods like cosine similarity maximize relevance at the cost of introducing redundant content, which can reduce information recall. To address this, we introduce **Diversity-Focused Retrieval-Augmented Generation (DF-RAG)** that systematically incorporates diversity into the retrieval step to improve performance on complex, reasoning-intensive QA benchmarks. DF-RAG builds upon the Maximal Marginal Relevance framework to select information chunks that are both relevant to the query and maximally dissimilar from each other. A key innovation of DF-RAG is its ability to optimize the level of diversity for each query dynamically at test time without requiring any additional fine-tuning or prior information. We show that DF-RAG improves F_1 performance on reasoning-intensive QA benchmarks by 4–10% over vanilla RAG using cosine similarity and also outperforms other established baselines. Furthermore, we estimate an Oracle ceiling of up to 18% absolute F_1 gains over vanilla RAG, of which DF-RAG captures up to 91.3%.

1 Introduction

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Guu et al., 2020) is a common technique for grounding Large Language Models' (LLMs) outputs in external knowledge due to its lightweight and composable architecture. However, RAG is often challenged by complex, reasoning-intensive question answering (QA) tasks as these questions involve retrieving and integrating relevant information scattered across multiple contexts (Li et al., 2025; Liu et al., 2024). One central limitation of common RAG implementations is its reliance on cosine similarity, which frequently over-selects semantically redundant content (Carbonell and Goldstein, 1998; Liu et al., 2025) and excludes complementary evidence that is crucial for performing complex reasoning (Asai et al., 2024; Xiong et al., 2021).

Fundamentally, for such complex, reasoning-intensive QA, a RAG system should only need to retrieve the few critical pieces of information required to bridge reasoning gaps to generate the right answer (Khattab et al., 2021). We are thus motivated to ask our first research question:

Can we improve our retrieval strategy to incorporate complementary evidence that current RAG approaches ignore such that it bridges reasoning gaps in complex QA tasks?

To answer this, we investigate the role of retrieval diversity in RAG pipelines for reasoning-intensive QA. We build upon Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998), a technique that balances query relevance with dissimilarity from selected information via a diversity parameter, λ. We hypothesize that strategically controlling retrieval diversity can bridge critical reasoning gaps required in complex reasoning QA and outperform current RAG baselines.

We first test this hypothesis by replacing cosine similarity with MMR using a fixed λ for all queries in a dataset. We find that the optimal λ varies across datasets, which suggests that individual queries within a dataset may benefit from tailored λ values. To estimate the upper bound of performance of dynamically selecting λ at the query-level, we design an Oracle that selects the optimal λ for each query using the ground truth. The performance of our Oracle helps us address our first research question: Oracle surpasses vanilla RAG by $\sim$18% and also outperforms full-context baselines while utilizing $\sim$15% of

the context. However, estimating the right λ at test time without any task-specific tuning is challenging. This leads to our second research question:

How can we design a RAG system capable of incorporating the optimal level of diversity at test time?

To this end, we introduce **Diversity-Focused Retrieval-Augmented Generation (DF-RAG)**, a RAG pipeline designed to approximate the Oracle's performance without access to the ground truth. DF-RAG adaptively determines the optimal λ for each query at test time through a multi-step process: (1) decomposing each query into individual steps with a planner LLM, (2) retrieving multiple candidate sets of chunks, each generated at predefined λ levels; and (3) selecting the chunk set with the highest potential to address the decomposed steps with an Executor LLM. DF-RAG does not require fine-tuning, utilizes a plug-and-play retrieval framework, and directly addresses our second question. By benchmarking DF-RAG across five QA datasets, we demonstrate that leveraging DF-RAG to find the optimal λ at test time can close the performance gap to the Oracle.

To empirically validate our systems for such complex reasoning-intensive QA tasks, we benchmark them on: 1) multi-hop QA datasets as they require reasoning over evidence scattered across multiple contexts and 2) non-multi-hop but challenging long context QA datasets as they still require reasoning over context to generate answers (Yang et al., 2018; Ho et al., 2020). Our empirical evaluation across $\sim$180 experiments shows that DF-RAG routinely outperforms strong baselines on multi-hop QA, achieving up to 9.5% gains over vanilla RAG across context lengths and closing as much as up to 91.3% of the gap to the Oracle. On non-multi-hop QA, DF-RAG achieves up to a 4% improvement over vanilla RAG and other baselines. To our knowledge, this is the first work to leverage query-level diversity to estimate Oracle performance bounds and to propose a RAG framework that narrows this gap by dynamically balancing diversity and relevance at test time.

2 Related Work

2.1 Multi-hop Question Answering

Early work on multi-hop QA established foundational benchmarks like HotpotQA (Yang et al., 2018) and MuSiQue (Trivedi et al., 2022), which require finding and reasoning over multiple supporting documents. Many methods perform the task by conducting inference over static or dynamic graphs to find reasoning paths (Fang et al., 2020; Ding et al., 2019; Zhang et al., 2021). Neural-based approaches have also attempted to decompose multi-hop questions into single-hop questions or generate follow-up questions based on retrieved information (Min et al., 2019; Cao et al., 2019; Malon and Bai, 2020). Recent advances in multi-hop QA include LLM-based approaches that leverage strong natural language understanding capabilities (Xu et al., 2021) through reasoning chain generation (Zelikman et al., 2022), question decomposition (Wu et al., 2024), and graph construction (Li and Du, 2023).

2.2 Retrieval Augmented Generation

Despite the superior language understanding and generation capabilities of LLMs, their performance remains limited on various downstream tasks due to challenges, such as outdated and long-tail knowledge (He et al., 2022; Kandpal et al., 2023), hallucination (Zuccon et al., 2023) and the lack of domain-specific knowledge (Shen et al., 2023). RAG (Guu et al., 2020; Lewis et al., 2020) has become the most widely adopted approach to address these issues by grounding LLM outputs in external knowledge for domain-specific downstream tasks. RAG pipelines operate by retrieving the most relevant document segments for a given query, and their effectiveness is therefore strongly influenced by the chunking strategies employed during retrieval. Recent advances in RAG include methods for reducing noise from chunks (Yan et al., 2024; Zhuang et al., 2023), chunk-free strategies to mitigate semantic loss (Qian et al., 2024), hierarchical arrangements of chunks via summarization techniques to capture global context (Sarthi et al., 2024), adaptive chunk-length strategies (Li et al., 2024), and approaches that extract global cues to generate supporting questions for improved answers (Zhao et al., 2024).

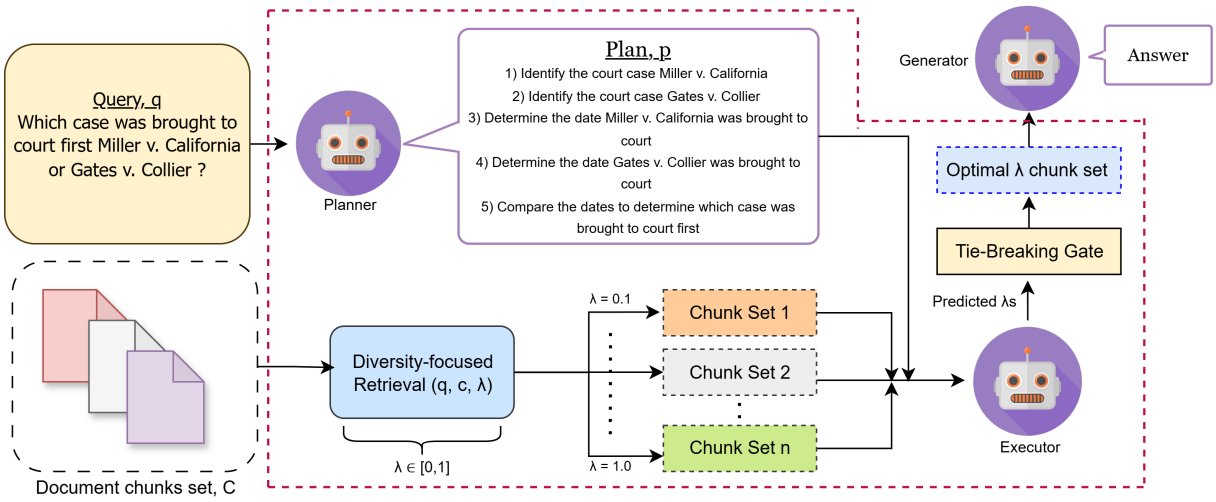


Figure 1: Architecture of DF-RAG: A training-free framework that dynamically selects the right level of diversity to generate answers for reasoning-intensive QA tasks. DF-RAG adaptively adjusts diversity for each query by utilizing the Planner, Executor and the Diversity-focused Retrieval. We illustrate this with an example query from HotpotQA (Yang et al., 2018).

2.3 Diversity in Information Retrieval

Traditionally, information retrieval has primarily emphasized relevance. Early work introduced Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998), which balances relevance and novelty. This inspired intent-aware ranking models aimed at improving subtopic coverage and evaluation (Agrawal et al., 2009; Clarke et al., 2008). Some approaches have incorporated explicit diversification objectives into learning-to-rank frameworks, jointly optimizing for relevance and subtopic coverage (Santos et al., 2015). Recent efforts have also explored diversity in RAG-based pipeline to reduce redundant evidence for downstream taks (Zhang et al., 2025; Wang et al., 2025). While these RAG pipelines highlight the importance of diversity, they either depend on additional fine-tuning or lack mechanisms to adaptively determine the optimal diversity, limiting their generalizability and effectiveness in practice.

3 Methodology

3.1 Diversity-focused Retrieval

To reduce content redundancy, we design a diversity-focused retrieval function built on MMR (Carbonell and Goldstein, 1998). Given a query q, a candidate chunk $c \in \mathcal{C}$, and a set of previously selected chunks $S = \{c_1, c_2, \dots\}$, the retrieval function assigns a geometric MMR score (gMMR) to c as follows:

$$\text{gMMR}(c) = \lambda \cos(\vec{q}, \vec{c}) + (1-\lambda)\sqrt{2 - 2\cos(\vec{c}, \vec{c}_S)} \quad (1)$$

Where the hyperparameter λ modulates the trade-off between relevance and diversity, and $\vec{c}_S$ is the centroid of our selected set of chunks so far, calculated by:

$$\vec{c}_S \triangleq \frac{1}{|S|} \sum_{s \in S} \vec{s} \quad (2)$$

To measure diversity, we replace standard cosine similarity with the Euclidean distance between the normalized embeddings, calculated as $\sqrt{2 - 2\cos(\vec{c}, \vec{c}_S)}$. This approach offers two advantages. First, it grounds diversity in a geometric framework enabling us to frame diversity as the spatial distance between a candidate chunk ($\vec{c}$) and the centroid of the already selected chunks ($\vec{c}_S$). Second, adopting this distance-based variant of MMR yields improved empirical performance in our downstream tasks. A detailed comparison against classical MMR is provided in Appendix B.2.

We greedily select the chunk c^* at the k-th step using the following conditions:

$$c^* = \begin{cases} \arg\max_{c \in \mathcal{C}} \cos(\vec{q}, \vec{c}) & \text{if } S = \varnothing, \\ \arg\max_{c \in \mathcal{C} \setminus S} \text{gMMR}(c) & \text{otherwise.} \end{cases} \quad (3)$$

3.2 Baseline: RAG with λ-Fixed gMMR

We implement a baseline using RAG with our gMMR function (Equation 1). In this approach, a constant λ is applied uniformly to all queries in a given benchmark. We vary $\lambda \in [0, 1]$ in increments of 0.1 to identify the value that yields the highest F_1 score for each dataset.

3.3 Oracle: A Theoretical Upper Bound for Query-Adaptive gMMR

We introduce an Oracle to represent the hypothetical upper bound for our query-adaptive approach. This Oracle is designed to determine the ideal gMMR diversity parameter λ for each query by leveraging the ground truth. This is achieved by sampling values from 0 to 1 in increments of 0.1 and selecting the λs whose corresponding chunk set yields the highest F_1 score with respect to the ground truth.

3.4 The DF-RAG Architecture

The Oracle establishes a theoretical performance upper bound for our query-adaptive gMMR. We next introduce DF-RAG, an architecture designed to approximate Oracle at test time for each query. As depicted in Figure 1, the architecture of DF-RAG integrates a plug-and-play retrieval component (highlighted in the purple dashed box) that uses an LLM-based Planner and an Executor for selecting the optimal λ for each query. When a query, q, is received, two processes are initiated in parallel: plan generation via the Planner as well as candidate chunk set retrieval using gMMR. Once the plan, P, and the candidate chunk sets are available, the Executor evaluates and selects the optimal chunk set thereby determining the optimal λ. Finally, given the selected chunk set, the Generator synthesizes a final answer to the query. These components are explained in more details below.

Planner for Step Decomposition: Given a query, q, the Planner generates an explicit sequence of steps $\{p_1, p_2, \ldots p_m\}$ that outline how the query should be solved, which we denote as a plan, P. Our hypothesis is that any candidate chunk set S sufficient to answer q should also be sufficient to execute P with the collective information it holds. Therefore, we assess the quality of a given S based on how well it supports P to make an informed prediction of λ at test time. The Planner operates under a one-shot prompting setup, with the prompting details detailed in Appendix A.1.

Candidate Chunk Sets Retrieval with gMMR: We retrieve candidate chunk sets, $S = \{c_1, c_2, \ldots, c_k\}$, from the document set, $\mathcal{C}$, using gMMR (Section 3.1). For our experiments, we sample chunks uniformly with $\lambda \in [0.1, 1]$ with a step size of 0.1. We exclude sampling a chunk set at $\lambda = 0$ because selecting chunks $2 \ldots k$ at $\lambda = 0$ disregards query relevance and introduces noisy information. The computational complexity of DF-RAG depends directly on the sampling strategy and uniform sampling yields a $O(n)$ complexity. Additionally, we propose a simpler binary-search-based sampling strategy that achieves $O(\log(n))$ complexity with comparable performance. More details and corresponding results of binary-search based sampling strategy for DF-RAG is presented in Appendix A.2.

Executor for Optimal Diversity Selection: Given a pool of candidate chunk sets retrieved using different λs, and a plan by the Planner, $P = \{p_i\}_{i=1}^{m}$, for the query, q, the Executor evaluates each candidate chunk set with respect to P. For each item $p_i \in P$, the Executor assigns a support score $r(p_i) \in \{0, \ldots, 5\}$, where 0 indicates no supporting information is present and 5 indicates direct evidence satisfying that step in the candidate chunk set. The overall score, denoted by $Retreival\ Support$, of a chunk set retrieved under λ is then computed as:

$$Retreival\ Support(\lambda, P) = \sum_{i=1}^{m} r(p_i) \quad (4)$$

The optimal query-specific λ, denoted λ^*, is then selected as follows:

$$\lambda^* = \arg\max_{\lambda \in \Lambda} Retreival\ Support(\lambda, P) \quad (5)$$

The Executor utilizes a few-shot prompting approach and the prompting details and the specific few-shot examples are discussed in Appendix A.1.

Tie-Breaking Gate: DF-RAG's Executor's assessment may identify multiple λs as equally optimal for a given query, but the system must select a single λ and its corresponding chunk set for answer generation due to the limit on context length we impose. To handle this, we introduce a Tie-Breaking Gate that selects one chunk set from the set of tied candidates. Our tie-breaking mechanism is deliberately lightweight and operates by selecting the median λ (upper median in case of even number of selected values) among the tied candidates.

Dataset	No. of Instances	Avg. Length
LongBench (Bai et al., 2024)		
HotpotQA	200	9,151
2WikiMQA	200	4,887
MuSiQue	200	11,214
MultifieldQA	150	4,559
∞*Bench* (Zhang et al., 2024)		
EN.QA	351	192,600

Table 1: Total number of queries and average length of contexts in number of words for each QA dataset.

Generator for Final Answer Synthesis: Similar to standard RAG systems, the final component is an LLM-based Generator. It is tasked with producing an answer given the original query q and the optimally-diverse chunk set c^* retrieved using λ^*. We follow the prompt released in LongBench (Bai et al., 2024) for answer generation, which is further detailed in Appendix A.1.

3.5 Experimental Details

Preliminaries: We perform comprehensive experiments using Qwen 2.5 72B (Yang et al., 2025) and Llama 3.3 70B (Grattafiori et al., 2024) as backbone LLMs. We use 16-bit floating precision of both these models and run our experiments on 2 NVIDIA A100 80GB GPUs. We perform retrieval based on sentence embeddings generated by 'multi-qa-mpnet-base-cos-v1'. For our proposed experiments (other than established baselines), we always retrieve 5 chunks where chunks are generated at the word level. We vary context lengths by varying the number of words, $w \in \{100, 200, 300\}$, corresponding to context lengths of approximately 500, 1000, and 1500 words. All our results are based on single-run experiments.

RAG with λ-Fixed gMMR: We benchmark RAG with gMMR against vanilla RAG to highlight the impact of diversity-focused retrieval. Both models use the same generator and differ only in their chunk selection strategy. Vanilla RAG selects the top 5 chunks based on cosine similarity of their embeddings, whereas RAG with λ-Fixed gMMR selects chunks using gMMR at fixed λ values across datasets.

Oracle Upper Bound: We benchmark our Oracle against vanilla RAG, RAG with λ-Fixed gMMR, and a long-context baseline. The long-context baseline (denoted as Long Context) is implemented by concatenating the entire context with the input query and feeding it to the LLM generator. This baseline is only applicable when the full context fits within the model's token limit, which excludes En.QA from long-context evaluation.

DF-RAG Baselines: We implement two DF-RAG variants: (1) DF-RAG , where median λ is the tie-breaking gate used and (2) DF-RAG + incremental context, denoted by IC. The IC serves as a condensed summary added to the Generator's context. The IC comprises the plan produced by the Planner and the short answer generated by the Executor when executing that plan. The motivation and analysis of ICs benefits are further detailed in Appendix C.3. We compare our DF-RAG variants to the Oracle and four baselines. The four baselines are (1) RAG with λ-Fixed gMMR, Vanilla RAG, and RAPTOR (Sarthi et al., 2024), and LongRAG (Zhao et al., 2024). To experiment with RAPTOR and LongRAG, we replace the backbone LLM while keeping all other implementation details consistent with their published setups. For LongRAG, we employ both the filter and extractor modules, as this configuration yields the best performance in their reported results. We benchmark against LongRAG and RAPTOR given their demonstrated success in prior work and their strategic designs for retrieving critical information. Both methods excel at complex reasoning tasks, making them well-suited as baselines.

Datasets: We benchmark our baselines, Oracle and DF-RAG on multiple reasoning-intensive QA datasets. Following prior work (Zhao et al., 2024; Li et al., 2024), we select datasets that (a) are in English, (b) contain queries that require reasoning, (c) are structured for QA evaluation. To this end, our benchmark suite includes three multi-hop QA datasets from LongBench (Bai et al., 2024) — HotpotQA, MuSiQue, and 2WikiMultihopQA — one non–multi-hop QA dataset from LongBench, MultifieldQA (English version), and one dataset from ∞Bench (Zhang et al., 2024), En.QA. Together, these datasets cover diverse domains, including Wikipedia articles, meetings, narratives, and research papers, making it challenging to reason over the context when answering the queries. Table 1 presents the datasets' statistics.

Evaluation Metrics: We follow the official metrics of LongBench and ∞Bench for the selected benchmarks, reporting answer-level F_1 scores against the ground truth.

Experiment Type	Llama 3.3 70B					Qwen 2.5 72B				
	HotpotQA	MuSiQue	2WikiMQA	MultifieldQA	En.QA	HotpotQA	MuSiQue	2WikiMQA	MultifieldQA	En.QA
context length = 500 words										
Oracle	69.5	46.2	65.3	54.9	41.3	67.7	43.8	64.8	57.2	40.1
Vanilla RAG	51.1	29.9	49.8	47.7	26.3	51.2	24.8	47.7	47.5	26.1
RAG with gMMR	53.3	30.9	54.8	48.2	27.6	55.0	27.6	52.7	49.7	27.7
DF-RAG	**58.1**	33.3	**58.1**	**48.3**	**29.3**	58.5	31.3	53.7	**51.0**	**28.2**
DF-RAG + IC	56.8	**34.0**	55.2	47.0	27.5	**60.1**	**32.7**	**56.8**	48.8	27.7
context length = 1000 words										
Oracle	68.6	49.7	68.6	56.2	45.2	67.5	48.7	67.2	56.7	44.1
Vanilla RAG	56.0	36.0	58.2	50.0	28.3	55.9	31.9	55.0	50.6	26.9
RAG with gMMR	56.0	37.5	58.7	50.9	30.1	56.7	**33.3**	55.5	**51.9**	30.3
DF-RAG	**60.3**	36.3	58.9	**51.6**	**31.5**	60.4	28.8	57.4	50.4	**31.1**
DF-RAG + IC	58.4	**38.6**	**63.3**	48.6	29.3	**62.1**	31.7	**59.7**	49.3	29.6
context length = 1500 words										
Oracle	69.5	51.9	74.9	56.1	47.8	68.7	48.6	69.9	55.9	47.4
Vanilla RAG	56.1	35.4	61.6	50.4	33.1	57.9	33.6	59.5	48.1	31.3
RAG with gMMR	58.1	35.9	63.4	**50.9**	**33.2**	60.1	**34.6**	59.8	**50.1**	32.5
DF-RAG	59.6	37.9	63.9	50.6	33.1	**61.8**	32.2	62.1	50.0	**34.0**
DF-RAG + IC	**61.8**	**38.0**	**68.6**	46.5	31.8	60.9	33.3	**69.0**	48.7	33.2
context length independent baselines										
LongRAG	57.4	38.1	61.7	**52.9**	22.9	57.7	36.8	59.9	47.8	22.4
RAPTOR	54.5	34.3	58.9	48.3	X	53.0	29.6	58.4	49.6	X

Table 2: F_1 scores of vanilla RAG and DF-RAG on five QA datasets across varying context lengths, evaluated using Llama 3.3 70B and Qwen 2.5 72B. Context length is varied by changing chunk size $w \in \{100, 200, 300\}$ words. **Bold** and underlined results indicate first and second in F_1 score. Results marked with X could not be performed due to memory limit restrictions.

4 Experimental Results

4.1 RAG with λ-Fixed gMMR

Our results revealed that RAG with λ-Fixed gMMR outperforms vanilla RAG (see Table 2 and Figure 2). We also found that numerous distinct dataset-level λ values outperformed vanilla RAG (see Figure 5). This helped to confirm that diversity-focused methods are beneficial, motivating our investigation into whether an optimal λ could be determined at the query-level. Details of the performance trends for each of these benchmarks across λ is further detailed in Appendix B.1.

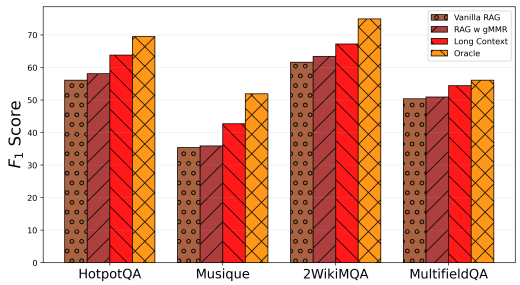


Figure 2: F_1 scores of vanilla RAG, RAG with gMMR, Long Context, and Oracle across LongBench benchmarks. Vanilla RAG, RAG with gMMR and Oracle operate at 1500 word context length. Results on En.QA are not reported as its context length exceeds our context limit.

4.2 Oracle: Upper Bound for Query-Adaptive gMMR

Our Oracle consistently outperforms both vanilla RAG and RAG with λ-Fixed gMMR across all benchmarks (see Figure 2). When compared to vanilla RAG, we observe consistent gains of $\sim$7–19% F_1 points with the largest gains observed for multi-hop QA benchmarks. A similar behavior is also observed as the Oracle beats RAG with λ-Fixed gMMR by $\sim 6 - 16\%$. Furthermore, the Oracle also outperforms Long Context by up to 9.2%, even when using up to $\sim 70 - 87\%$ less context (see Figure 2). This demonstrates that query-aware, optimally-tuned retrieval diversity can be more effective than processing the entire context, provided we find the right λ. As the Oracle achieves the highest performance among all baselines, it serves as DF-RAG's performance target. Accordingly, a dashed line follows Oracle's score in Table 2.

4.3 DF-RAG

DF-RAG beats RAG with λ-Fixed gMMR. We observe that DF-RAG beats RAG with λ-Fixed gMMR consistently in various benchmarks across context lengths. Specifically, DF-RAG outperforms RAG with λ-Fixed gMMR with gains ranging from $\sim 2 - 10\%$ across context lengths and models (see Figure Table 2). We generally ob-

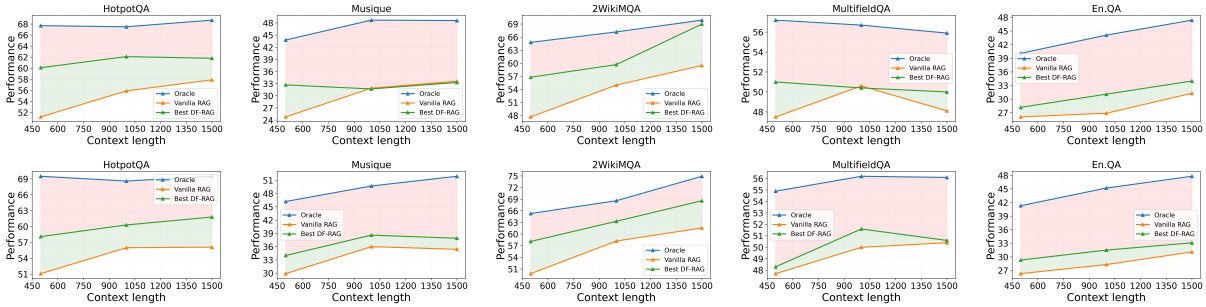


Figure 3: Performance (F_1 score) curves, of Oracle, vanilla RAG, and best DF–RAG across context lengths on all five QA benchmarks. The first row shows Qwen 2.5 72B as the backbone LLM whereas the second row shows Llama 3.3 70B.

serve larger gains with DF-RAG over RAG with λ-Fixed gMMR at longer context length settings (1500 words), suggesting that DF-RAG's ability of successfully picking the right λ for each query helps longer context settings, as it gives more flexibility at retrieving information. However, we also observe that DF-RAG is outperformed by RAG with λ-Fixed gMMR in a small number of cases (MuSiQue and MultifieldQA datasets), but the magnitude of drop in performance is very minimal. All together, the performance shows DF-RAG's success and importance of its ability to accurately predict optimal λ at test time.

DF-RAG outperforms various RAG baselines. Across benchmarks, DF-RAG consistently outperforms vanilla RAG, achieving up to +8.9% absolute improvement at the 500-word context length, and up to +6.2% and +9.5% at the 1000- and 1500-word settings, respectively. DF-RAG maintains a clear advantage over vanilla RAG across all benchmarks, with more pronounced gains on HotpotQA and 2WikiMQA (see Figure 3). Beyond vanilla RAG, DF-RAG also outperforms LongRAG and RAPTOR, two context length–independent baselines (LongRAG leverages global cues, while RAPTOR recursively summarizes the full context). Their results are shown in the last two rows of Table 2. Notably, at the 1500-word context length, DF-RAG surpasses RAPTOR across all benchmarks and model families, and similarly outperforms LongRAG on every benchmark except MultifieldQA with Llama 3.3 70B and MuSiQue with Qwen 2.5 72B.

DF-RAG approaches Oracle performance. Beyond outperforming baselines across benchmarks, DF-RAG narrows the gap to the Oracle, revealing its potential across datasets, model families, and context lengths. To measure its effectiveness, we calculated the portion of the performance gap between vanilla RAG and the Oracle that DF-RAG closes. On multi-hop QA benchmarks, DF-RAG closes an average of 41.5%, 32.1%, and 47.7% of this gap at 500, 1000, and 1500-word context lengths, respectively (see Figure 3 and Table 2). For non–multi-hop QA benchmarks, DF-RAG covers on average ∼12-20% of the gap. These results highlight two key insights: (1) DF-RAG effectively bridges reasoning gaps in multi-hop QA by adaptively balancing relevance and diversity through its retrieval component as it approaches Oracle performance (analyzed in Appendix D.3). In one case, DF-RAG covers up to 91.3% of the gap on a particular benchmark (2WikiMQA for Qwen 2.5 72B at 1500-word context length). (2) DF-RAG also maintains strong performance on complex non–multi-hop QA tasks, too (just below around 5% of the Oracle for MultifieldQA). Together, the performance comparison of DF-RAG with the Oracle showcases DF-RAG's ability to adaptively predict optimal λ at test time. Further analysis of DF-RAG's success and failure cases are presented in Appendix D.

DF-RAG can leverage parallelization. To assess the computational efficiency of DF-RAG, we compare it against our reported RAG baselines along two key dimensions: tokens processed and wall-clock latency (see Table 3). We evaluate all methods on 20 randomly selected samples from MultifieldQA, which features reasonably long contexts. Both DF-RAG and vanilla RAG operate with 300-word chunks. While DF-RAG processes the largest number of tokens among the baselines on this dataset, DF-RAG's token usage has a constant upper bound, unlike other methods whose token consumption scales with context length and structure. DF-RAG's latency is significantly lower than RAPTOR and LongRAG (see Table 3), as its most

Method	Tokens Processed	Latency
Vanilla RAG	$\sim 1,500$	5s
RAPTOR	$\sim 10,000$	326s
LongRAG	$\sim 14,000$	127s
DF-RAG	$\sim 15,000$	27s

Table 3: Computational characteristics of different RAG methods, comparing total tokens processed and wall-clock latency averaged across 20 samples in MultifieldQA.

time-consuming component – the Executor – can parallelize its independent operations. To visualize the benefits of parallelization in our DF-RAG experiments, we reduce the number of Executor LLM calls to one, as parallelized calls incur the same latency as a single call.

5 Conclusion

In this work, we addressed the challenges RAG pipelines face on complex, reasoning-intensive QA tasks, identifying their reliance on cosine similarity as a key limitation that leads to redundant contexts. To overcome this, we introduced a retrieval function that balances diversity and relevance by a control parameter – λ. We incorporated this retrieval function at a dataset-level which showed performance improvements over vanilla RAG. With our given retrieval mechanism, we then designed an Oracle to understand the upper bound of performance improvement possible by controlling our λ at a query level, which revealed up to 18% absolute performance gains from vanilla RAG. Building on these insights, we proposed DF-RAG, which employs query-aware diversity by predicting the optimal λ for each query at test time without prior task-specific tuning. To test our system's performance on complex reasoning-intensive QA tasks, we evaluated DF-RAG on multi-hop QA and long-context non–multi-hop QA datasets across various context lengths and 2 model families. DF-RAG routinely outperforms strong baselines across context lengths and recovers up to 91.3% of the performance gap between vanilla RAG and the Oracle. Furthermore, complexity wise, DF-RAG uses constant token processing and can achieve faster execution through parallelization, compared to our baselines. Our results highlight the importance of diversity-focused retrieval for improving RAG pipelines on QA tasks and direct future directions for further innovation at closing the gap to Oracle performance.

Limitations

While DF-RAG brings about performance improvements, there are limitations in our approach that merit discussion. First, given the large number of experiments conducted (over 180 in total), we only benchmarked using two LLM families (Qwen and Llama). Second, while we benchmarked on 1,101 samples across five datasets from LongBench and ∞Bench, leveraging a wider range of datasets with more samples from such unified suites would enable stronger assessments of generalizability. Third, DF-RAG could benefit from additional experimentation with other notable baselines in the literature, e.g Self-RAG (Asai et al., 2024) and GraphRAG (Edge et al., 2025). Finally, a central component of DF-RAG is its Planner and Executor, which rely on LLMs. While we conduct systematic ablation studies on both these components to evaluate their performances (see Appendix C.1 and C.2), improved prompting and other test time strategies (Snell et al., 2024) may further improve their effectiveness and raise the ceiling of DF-RAG.

Ethical Considerations

Our research underwent a standard internal review. We used publicly available datasets and LLMs in accordance with their intended scope, and we refer readers to the original dataset papers for details on anonymization, preprocessing, and annotator information. All artifacts (existing baselines and benchmarks) used in this this paper are available under their respective open-source licenses (except for LongRAG; we only reproduce their code). Our experiments focus exclusively on the English language; however, the proposed methodology makes no assumptions about the language of operation and we encourage future work to explore the applicability of our approach in multilingual settings. Furthermore, we acknowledge the propagation of various biases through LLM-based systems and while our methods do not study the implications of biases in our system, we recommend that any system implementing our methodology undergo rigorous testing of (1) the documents retrieved to prevent the propagation of biases, and (2) the models used to generate answers to ensure they do not promote biases during generation. Such systems should also be subject to continuous monitoring. We also acknowledge the use of ChatGPT for fixing grammatical and spelling errors.

References

Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM)*, pages 5–14. ACM.

Akari Asai, Xinyan Wu, Sewon Min, Wenhu Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. LongBench: A bilingual, multi-task benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Yu Cao, Meng Fang, and Dacheng Tao. 2019. BAG: Bi-directional attention entity graph convolutional network for multi-hop reasoning question answering. 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 357–362, Minneapolis, Minnesota. Association for Computational Linguistics.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, and 1 others. 2024. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*.

Charles L. A. Clarke, Mounia Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 659–666. ACM.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2694–2703, Florence, Italy. Association for Computational Linguistics.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2025. From local to global: A graph rag approach to query-focused summarization. *Preprint*, arXiv:2404.16130.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Thinkless: Llm learns when to think. *arXiv preprint arXiv:2505.13379*.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2020. Hierarchical graph network for multi-hop question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8823–8838, Online. 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, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Michael Hassid, Gabriel Synnaeve, Yossi Adi, and Roy Schwartz. 2025. Don't overthink it. preferring shorter thinking chains for improved llm reasoning. *arXiv preprint arXiv:2505.17813*.

Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*.

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).

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 15696–15707. PMLR.

Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Baleen: robust multi-hop reasoning at scale via condensed retrieval. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA. Curran Associates Inc.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020.

Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*.

Ruosen Li and Xinya Du. 2023. Leveraging structured information for explainable multi-hop question answering and reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6779–6789, Singapore. Association for Computational Linguistics.

Zhuoqun Li, Xuanang Chen, Haiyang Yu, Hongyu Lin, Yaojie Lu, Qiaoyu Tang, Fei Huang, Xianpei Han, Le Sun, and Yongbin Li. 2025. StructRAG: Boosting knowledge intensive reasoning of LLMs via inference-time hybrid information structurization. In *The Thirteenth International Conference on Learning Representations*.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024. Retrieval augmented generation or long-context LLMs? a comprehensive study and hybrid approach. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 881–893, Miami, Florida, US. Association for Computational Linguistics.

Jingyu Liu, Jiaen Lin, and Yong Liu. 2024. How much can rag help the reasoning of llm? *Preprint*, arXiv:2410.02338.

Tianyu Liu, Jirui Qi, Paul He, Arianna Bisazza, Mrinmaya Sachan, and Ryan Cotterell. 2025. Pointwise mutual information as a performance gauge for retrieval-augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1628–1647, Albuquerque, New Mexico. Association for Computational Linguistics.

Christopher Malon and Bing Bai. 2020. Generating followup questions for interpretable multi-hop question answering. *Preprint*, arXiv:2002.12344.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6097–6109, Florence, Italy. Association for Computational Linguistics.

Hongjin Qian, Zheng Liu, Kelong Mao, Yujia Zhou, and Zhicheng Dou. 2024. Grounding language model with chunking-free in-context retrieval. *arXiv preprint arXiv:2402.09760*.

Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. 2015. Search result diversification. *Foundations and Trends® in Information Retrieval*, 9(1):1–90.

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*.

Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2023. In chatgpt we trust? measuring and characterizing the reliability of chatgpt. *arXiv preprint arXiv:2304.08979*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

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.

Zhichao Wang, Bin Bi, Yanqi Luo, Sitaram Asur, and Claire Na Cheng. 2025. Diversity enhances an llm's performance in rag and long-context task. *Preprint*, arXiv:2502.09017.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Jian Wu, Linyi Yang, Yuliang Ji, Wenhao Huang, Börje F. Karlsson, and Manabu Okumura. 2024. Gendec: A robust generative question-decomposition method for multi-hop reasoning. *Preprint*, arXiv:2402.11166.

Wenhan Xiong, Payal Bajaj, Zijian Yu, Zhuyun Guo, Scott Wen-tau Song, Daniel Campos, Nick Craswell, Rajan Majumder, and Saurabh Tiwary. 2021. Answering complex open-domain questions with multi-hop dense retrieval. In *9th International Conference on Learning Representations (ICLR)*.

Weiwen Xu, Yang Deng, Huihui Zhang, Deng Cai, and Wai Lam. 2021. Exploiting reasoning chains for multi-hop science question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1143–1156, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*.

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, Junyang Lin, Kai Dang, and 23 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

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.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. 2022. Star: self-taught reasoner bootstrapping reasoning with reasoning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024. ∞Bench: Extending long context evaluation beyond 100K tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15262–15277, Bangkok, Thailand. Association for Computational Linguistics.

Yuyu Zhang, Ping Nie, Arun Ramamurthy, and Le Song. 2021. Answering any-hop open-domain questions with iterative document reranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 481–490, New York, NY, USA. Association for Computing Machinery.

Zhange Zhang, Yuqing Ma, Yulong Wang, Shan He, Tianbo Wang, Siqi He, Jiakai Wang, and Xianglong Liu. 2025. Lexical diversity-aware relevance assessment for retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27758–27781, Vienna, Austria. Association for Computational Linguistics.

Qingfei Zhao, Ruobing Wang, Yukuo Cen, Daren Zha, Shicheng Tan, Yuxiao Dong, and Jie Tang. 2024. LongRAG: A dual-perspective retrieval-augmented generation paradigm for long-context question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22600–22632, Miami, Florida, USA. Association for Computational Linguistics.

Shengyao Zhuang, Bing Liu, Bevan Koopman, and Guido Zuccon. 2023. Open-source large language models are strong zero-shot query likelihood models for document ranking. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8807–8817, Singapore. Association for Computational Linguistics.

Guido Zuccon, Bevan Koopman, and Razia Shaik. 2023. Chatgpt hallucinates when attributing answers. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, SIGIR-AP '23, page 46–51, New York, NY, USA. Association for Computing Machinery.

A Methods

A.1 System Prompts

We release all our system prompts starting from the Planner, Executor and the Generator. Figure 6 shows the prompt used for the Planner in DF-RAG. Figure 7 shows the prompt used by our Executor. The Executor utilizes few shot prompting and the few shot examples specifically utilized are further detailed in Figure 8 and Figure 9.

A.2 DF-RAG with Binary-Search sampling

DF-RAG's latency is primarily determined by the number of LLM calls made by the Executor, which is in turn contingent upon the sampling strategy used to sample chunk sets. To improve efficiency, we introduce a binary-search–based sampling strategy, motivated by its effectiveness in locating peaks in unimodal distributions. This approach reduces the sampling complexity from n to $\log n$. Binary-search sampling achieves comparable performance to uniform sampling across models (see Table 7), with only minimal drops in accuracy.

B Additional Experimental Results

B.1 RAG with λ-Fixed gMMR:

Figure 5 presents a detailed performance comparison between vanilla RAG and RAG with λ-Fixed gMMR across the five benchmarks at the different context lengths settings with Llama 3.3 70B as the backbone LLM. In vanilla RAG, performance is invariant to λ, producing a horizontal line, whereas RRAG with λ-Fixed gMMR's performance varies with λ, with each benchmark exhibiting a clear peak at a given context length. For each benchmark, we find at least one context length contains a λ where RAG with λ-Fixed gMMR outperforms vanilla RAG. At the 500-word context length, the benefits of using RAG with λ-Fixed gMMR is more prominent. whereas for 1000 and 1500-word context length settings, F_1 improvements are less substantial.

B.2 Performance comparison between Traditional MMR and our Diversity-focused Retrieval

We compare DF-RAG using traditional MMR with DF-RAG using our proposed gMMR retrieval function. Experiments are conducted with 500-word chunks and Llama 3.3 70B as the backbone LLM (see Table 4). DF-RAG with gMMR outperforms the traditional MMR variant on three out of five benchmarks, and closely matches its performance on the remaining two. Overall, gMMR yields higher average gains compared to traditional MMR.

Benchmarks	Traditional MMR	gMMR
HotpotQA	56.7	**58.1**
MuSiQue	**34.1**	34.0
2WikiMQA	53.6	**58.1**
MultifieldQA	**49.0**	48.3
En.QA	28.2	**29.3**

Table 4: F_1 scores of DF-RAG performing with Traditional MMR vs gMMR

C Ablation Study

C.1 Effectiveness of the Planner component

Since the Planner is responsible for breaking complex reasoning-intensive questions into intermediate steps, we evaluate its effectiveness based on the number of steps it generates per query. We hypothesize that an effective planner should avoid overplanning while breaking queries into a reasonable number of steps. Therefore, a good planner should produce steps close to the number of hops for multi-hop QA benchmarks. The Planner produces an average of 2.88 steps for 2WikiMQA, 2.71 for HotpotQA, and 2.81 for MuSiQue. These values closely align with the original number of hops in these datasets, supporting our claim that the Planner generates well-calibrated plans.

C.2 Effectiveness of the Executor component

We evaluate the effectiveness of the Executor in our DF-RAG pipeline by examining the distribution of its predicted λ values from the Executor across datasets. An effective Executor should assign a diverse range of λ values, reflecting its ability to adapt retrieval diversity to the unique requirements of each query. Such behavior would generate a smooth and broad distribution of λs. In contrast, a poorly performing Executor would produce a narrow, concentrated distribution – indicating that it defaults to similar λ values for most queries, failing to capture query-specific diversity needs and effectively outputting non-informative predictions. Figure 4 shows the distribution of λs from Executor and from our Oracle.

C.3 Effectiveness of appending the incremental context

Our results show that the DF-RAG variant outperforms the DF-RAG + IC variant in several settings – for example, on HotpotQA, 2WikiMQA and MultifieldQA and En.QA at the 500-word context length with Llama 3.3 70B; on HotpotQA, MultifieldQA and En.QA at 1500 words with Qwen 2.5 72B. The IC augmentation was intended to provide a concise rationale for chunk selection, which—following findings from Chain-of-Thought (CoT) (Wei et al., 2022)—should have enhanced performance. However, contrary to this expectation, our experiments reveal that in some cases (notably always on MultifieldQA and En.QA), appending the incremental context leads to slightly lower performance compared to omitting it. We attribute this finding to two factors: 1) For non-multi-hop reasoning, extended reasoning chains are not only computationally inefficient (Chen et al., 2024) but also potentially detrimental to performance (Hassid et al., 2025). Drawing a parallel to this "overthinking" problem, the IC could make less complicated queries seemingly complicated through excessive planning steps - deviating LLMs reasoning line of bridging information that eventually hurts performance. Consequently, mitigation strategies developed to curb model overthinking may also be applicable to DF-RAG (Fang et al., 2025). 2) Our experiments still use relatively small contexts (500-1500 words) where the information required to reach a multi-hop answer is already highly condensed, leaving little room for chain-of-thought reasoning to provide additional benefit.

D Error Analysis

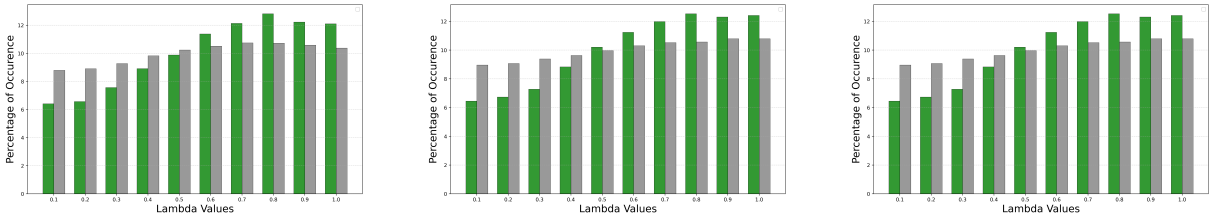
We analyze certain cases where DF-RAG both succeeds and fails and try to draw intuition about why such cases arise. In these cases, we compare DF-RAG with vanilla RAG to better understand the point of success and failure.

D.1 DF-RAG successful cases

In most successful cases, we find that DF-RAG is able to retrieve information that vanilla RAG overlooks. This often occurs when DF-RAG captures chunks that align with each reasoning step required to answer a multi-hop query by fetching diverse and complementary information, whereas vanilla RAG often misses key bridging information because such chunks have low surface similarity to the query and are therefore excluded by cosine-similarity–based retrieval. An illustrative example is provided in Table 5. For the input query *What is the place of birth of the performer of the song Ruleta (Inna Song)?*, DF-RAG retrieves both complementary pieces of information: *Ruleta is recorded by Romanian singer Inna* and *Inna Elena Alexandra Apostoleanu was born in Mangalia* (highled in the example). In contrast, vanilla RAG only retrieves the first piece of information about *Ruleta being recorded by Inna*, but not the birthplace information. Vanilla RAG retrieves many chunks that mostly discuss Ruleta because they are highly relevant to the query, but this displaces the other information needed to make the final reasoning step and generate the answer, i.e., *Inna being born in Mangalia*.

D.2 DF-RAG failure cases

Our analysis to visualize certain cases where DF-RAG fails revealed that DF-RAG sometiems fails to retrieve information that is needed for reasoning over multi-hop QAs, and this is a limitation common to RAG that stems from the inherent nature of information loss in retrieval. However, there are also cases where key information is retrieved appropriately, but due to the structure of the chunks presented, the Generator finds it difficult to reason effectively to generate the answer (see Table 6). For the question *What is the place of birth of the director of the film Ninamaninja Kalpadukal?*, it is important first to find *who the director of Ninamaninja Kalpadukal is* and second *where his birthplace is*. If we look at the chunks retrieved by both DF-RAG and vanilla RAG, we can see that both pieces of information are retrieved by the systems (highlighted in the examples). However, it is important how the information is presented to the generator. For the chunks retrieved by DF-RAG, the chunk that talks about the village *Methala* appears somewhat abruptly, making it difficult to decipher that Methala is the birthplace of the identified director of Ninamaninja Kalpadukal. By contrast, in the chunks retrieved by vanilla RAG, the information required for performing the second reasoning step comes right after introducing the director of Ninamaninja Kalpadukal, making it sound like the place where the director spent his early life, and thus the generator can bridge the gap to generate *Methala* as the answer.



(a) 500 word context length (b) 1000 word context length (c) 1500 word context length

Figure 4: Distribution of λ values from the Executor of DF-RAG (green) and the Oracle (gray). Both exhibit smooth distributions, though the Oracle's is slightly smoother, reflecting the source of DF-RAG's performance gap.

D.3 Understanding performance trends across benchmarks

Multi-hop QA benchmarks:

We observe that DF-RAG is able to close a substantial portion of the performance gap between vanilla RAG and our Oracle on two of the three multi-hop QA benchmarks—2WikiMQA and HotpotQA. However, the improvements are smaller on MuSiQue for DF-RAG compared to vanilla RAG. One reason for this discrepancy is the relative complexity of the benchmarks. HotpotQA predominantly contains 2-hop questions, where the reasoning chain involves fewer bridges and is thus easier compared to questions requiring more hops. For instance, in the question: *Prior to playing for Michigan State, Keith Nichol played football for a school located in what city?*, the reasoning needs to identify: 1) *Identify the school Keith Nichol played for before Michigan State* and 2) *Determine the city where that school is located*. Both of these questions can be answered solely by relying on the query as the entities of interested are directly placed there. By retrieving a sufficiently diverse set of information about *Keith Nichol*, we can find details about his school prior to Michigan State. Similarly, by retrieving information about *schools*, we can also identify the city in which the school is located.

In contrast, MuSiQue contains a greater number of hops per question. For example, the question *When was the last time Peter Till's sports team beat the winner of the 1894–95 FA Cup?* requires essentially three hops: 1) *Identify Peter Till's sports team*, 2) *Identify the winner of the 1894–95 FA Cup*, and 3) *Determine the last time the team identified in the first step defeated the winner of the 1894–95 FA Cup*. We can answer the first hop by relying on retrieving a sufficiently diverse retrieval set on Peter Till. We can also perform the second hop by retriving information about the winners of FA cup on 1894-1895. However the third hop needs connecting the entities we find in the first and second hop, which is impossible to guess without answering them and therefore retrieving information solely based on question provides minimal support. As a result, unlike in HotpotQA, simply relying on the question often provides limited guidance on what evidence needs to be retrieved to answer the final hop and hence both DF-RAG and Vanilla RAG fails. Moreover, MuSiQue mentions that for each question presented, it is compulsory to perform each hop as they filtered out questions where later hops could be answered without knowing the earlier answers, making no shortcut jumps possible and a difficult dataset to address at short context scenarios.

Non multi-hop QA benchmarks: DF-RAG also demonstrates improvements on MultifieldQA and En.QA. Although these datasets are not inherently multi-hop, their contexts often contain distracting information that appears relevant but leads models to incorrect answers. DF-RAG mitigates this by retrieving not only relevant but also diverse evidence, increasing the likelihood of capturing the true answer amid such noise. For example, in MultifieldQA, the question *What is the score achieved by the authors for Track-2?* often misleads vanilla RAG. The retriever selects a chunk stating *All of our submissions were the top submissions for each track, which surpassed the next best competitors by a margin of 4.5% and 5.6% for Track-1 and Track-2 respectively*, causing the model to output 5.6%, which is the margin of improvement rather than the score. In contrast, DF-RAG successfully retrieves the chunk *We achieve a score of 58.54% for Track-1 and 85.61% for Track-2*, which, despite surrounded with noisy information, provides the correct answer.

Question: What is the place of birth of the performer of song Ruleta (Inna Song)?
Selected chunks by DF-RAG: EDM and reggaeton-influenced track, containing touches of Indian and Caribbean music. When asked by a Direct Lyrics interviewer to describe the lyrics and sound of "Ruleta", Inna said, "It's summerish, it's cool, it's fun and I dare to say... it's catchy and addictive!" Reception Upon its release, "Ruleta" received positive reviews from music critics. Currinn from CelebMix wrote, "this new song has Inna proving that she's multilingual", and compared the style of "Ruleta" to her "Heaven" (2016). Kevin Apaza, writing for Direct Lyrics thought, "summer doesn't officially start until Romanian pop queen Inna releases her summer single", further calling it Passage 1: Inna Elena Alexandra Apostoleanu (born 16 October 1986), known professionally as Inna (stylized in all caps), is a Romanian singer and songwriter. Born in Mangalia and raised in Neptun, she studied political science at Ovidius University before meeting the Romanian trio Play & Win and pursuing a music career. She adopted the stage name "Alessandra" and a pop-rock style in 2008; later that year, she changed her stage name to "Inna" and began releasing house music. "Hot" (2008), her debut single, was a commercial success worldwide and topped the Romanian and Billboard's Hot Dance Airplay chart, among others. mtvU for his video that depicted a love affair between a woman and her television, with the TV containing the green-screened Decemberists video footage. Passage 3: Ruleta (Inna song) "Ruleta" (English: "Roulette") is a song recorded by Romanian singer Inna, featuring collaborative vocals by Romanian performer Erik, for her fifth studio album, Nirvana (2017) ...
Selected Chunks by Vanilla RAG: ... An editor of Bihoreanul praised the "attractive" rhythms of "Ruleta" and noted its potential commercial success in Romania.Commercially, the song debuted at number 32 on Romania's Airplay 100 for the week ending 9 July 2017 as the highest new entry, marking one of Inna's highest debuts on the chart. In late August 2017, "Ruleta" reached its peak position at number three, becoming the singer's highest charting single since "Bop Bop" (2015), which peaked at number two. The track also opened at number 39 on the French Club 40 chart for the week ending 15 July 2017 and rose up to number 26 on the following week. "Ruleta" further charted in other countries, most notably reaching number two in Bulgaria. Music video An accompanying music video for "Ruleta" was shot at the Domeniul Greaca resort in Romania by Barna Nemethi, with Marius Apopei serving as the director of photography. Inna recalled the experience, "Actually, I was super chill in that day, super relaxed, just going with the flow. Most of the times, I'm kind of stressed out, because I want everything to be perfect. Not too many funny stuff, just having fun."The clip was uploaded onto her official mtvU for his video that depicted a love affair between a woman and her television, with the TV containing the green-screened Decemberists video footage. Passage 3: Ruleta (Inna song) "Ruleta" (English: "Roulette") is a song recorded by Romanian singer Inna, featuring collaborative vocals by Romanian performer Erik, for her fifth studio album, Nirvana (2017). It was made available for digital download and streaming on 21 June 2017 by Global Records as the album's second single. The track was written by Inna, Erik, Marius Dia and Breyan Isaac, while production was handled by Sebastian Barac, Marcel Botezan and David Ciente.
Answer of DF-RAG: Mangalia
Vanilla RAG Answer: Romania
Golden Answer: Mangalia

Table 5: Selected chunks of a successful case by DF-RAG and Vanilla RAG. DF-RAG successfully retrieves the information needed to perform the 2nd hop to answer the query.

Question: What is the place of birth of the director of film Ninamaninja Kalpadukal?
Selected chunks by DF-RAG: an assistant director for Tamil - Telugu movies in Newton Studios, he later distinguished himself as a producer, director and scriptwriter. His directorial debut, Ninamaninja Kalpadukal, which portrays the trials the Indo-China war, won 4 awards including the President's silver medal for best regional film, and the award for The Best Director. He directed about 6 movies and produced 2 movies on his own. He wrote the screen play and directed 4 episodes of Aithihyamala for Doordarshan. He also directed a tele-film titled Kudajadri. Filmography Passage 4: Peter Levin Peter Levin is an American director of film, television and theatre. into a family known as "Kallil" in Methala, Near Kalady in Kerala, he completed his schooling in Paravoor and then was graduated in economics from Serhampur, Bengal. He was an avid reader always and appreciated the nuances of literature. His first short story was published in the weekly Prasanna Keralam from Kottayam when he was studying in high school. Since then, his stories have appeared in weeklies. Writing career One of his first of his short stories was published in the weekly Prasanna Keralam from Kottayam, when he was studying in high school.The editor of Kaumudi, K Balakrishnan, lead him Norwegian Museum of Cultural History. In 2010 he was decorated with the Royal Norwegian Order of St. Olav. Passage 8: S. N. Mathur S.N. Mathur was the Director of the Indian Intelligence Bureau between September 1975 and February 1980. He was also the Director General of Police in Punjab. Passage 9: Ian Barry (director) Ian Barry is an Australian director of film and TV. Select credits Waiting for Lucas (1973) (short) Stone (1974) (editor only) The Chain Reaction (1980) Whose Baby? (1986) (mini-series) Minnamurra (1989) Bodysurfer (1989) (mini-series) Ring of Scorpio (1990) (mini-series) Crimebroker (1993) Inferno (1998) (TV movie) Miss Passage 1: Ninamaninja Kalpadukal Ninamaninja Kalpadukal (Bloodstained Footprints) is a 1963 Malayalam language film, directed by N. N. Pisharody and produced by N.K. Karunakara Pillai and Shobhana Parameswaran Nair. The lead role is played by Prem Nazir, with Ambika, Sheela and Madhu, who debuted with this film. The film is based on a novel by Parappurath and portrays the trials the Indo-China war. It won the National Film Award for Best Feature Film in Malayalam. It was a super hit movie.There are many evergreen songs in the film, including ...
Selected Chunks by Vanilla RAG: an assistant director for Tamil - Telugu movies in Newton Studios, he later distinguished himself as a producer, director and scriptwriter. His directorial debut, Ninamaninja Kalpadukal, which portrays the trials the Indo-China war, won 4 awards including the President's silver medal for best regional film, and the award for The Best Director. He directed about 6 movies and produced 2 movies on his own. He wrote the screen play and directed 4 episodes of Aithihyamala for Doordarshan. He also directed a tele-film titled Kudajadri. Filmography Passage 4: Peter Levin Peter Levin is an American director of film, television and theatre. Passage 1: Ninamaninja Kalpadukal Ninamaninja Kalpadukal (Bloodstained Footprints) is a 1963 Malayalam language film, directed by N. N. Pisharody and produced by N.K. Karunakara Pillai and Shobhana Parameswaran Nair. The lead role is played by Prem Nazir, with Ambika, Sheela and Madhu, who debuted with this film. The film is based on a novel by Parappurath and portrays the trials the Indo-China war. It won the National Film Award for Best Feature Film in Malayalam. It was a super hit movie.There are many evergreen songs in the film, including "Mamalakalkkappurathu" (by P. B. Sreenivas) and "Anuraga Natakathil" (by K. P. into a family known as "Kallil" in Methala, Near Kalady in Kerala, he completed his schooling in Paravoor and then was graduated in economics from Serhampur, Bengal. He was an avid reader always and appreciated the nuances of literature. His first short story was published in the weekly Prasanna Keralam from Kottayam when he was studying in high school. Since then, his stories have appeared in weeklies...
Answer of DF-RAG: not specified in the provided passages.
Vanilla RAG Answer: methala, near kalady in kerala
Golden Answer: Methala

Table 6: Selected chunks of a failure case by DF-RAG and Vanilla RAG. DF-RAG successfully retrieves the information needed but fails to stitch the reasoning gap to answer the query.

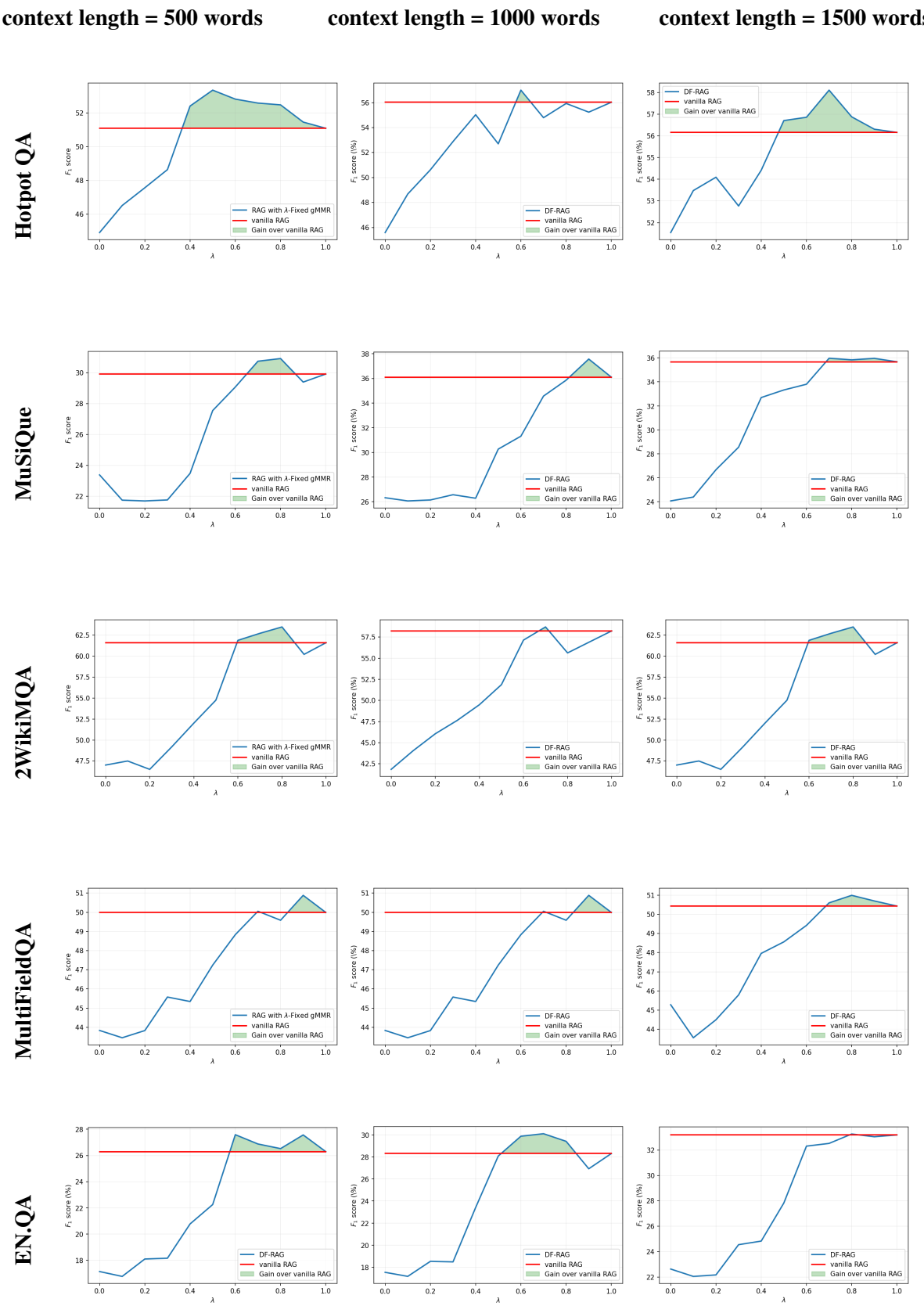


Figure 5: F_1 Scores comparison of RAG with λ-Fixed gMMR vs vanilla RAG at different chunk sizes for our 5 benchmarks utilizing Llama 3.3 70B as the backbone LLM.

Planner

Instruction: You are an expert question decomposer planner.

Task: Break the user's **single question** into the minimal ordered list of sub-questions that must all be answered to fully answer the original question harnessing your zero-shot capabilites. Follow the output format exactly as shown in the example below to generate a plan (a list of sub-questions).

Question: Where was the author of the book 'Jacob' Born?

Output Planner :
1) Identify the author of Jacob
2) Identify where the author was born

Only produce the PLAN section — no extra commentary for the following question:

Question: <Question>

Figure 6: Prompt used by the Planner

Executor

Instruction: You are a strict judge evaluating text chunks. Only count information that is explicitly stated.

Task: Score each question 0-5 based on how directly it can be answered:
5: EXPLICIT answer with specific facts/numbers stated directly
4: Clear and complete answer with direct relevant information present
3: Mostly complete answer, minor missing detail, but can be inferred
2: Partial information, key details missing
1: Very limited relevance, mostly vague
0: No relevant information or requires inference

STRICT RULES:
- Only count information DIRECTLY STATED in chunks
- Do NOT give points for logical leaps or inferences
- Do NOT give points for vague or tangential content

Your reasoning for the score values you provide should be within compact. **You MUST remember this.** At the end you must answer in the format: Total Score: <Total Score>
Follow the examples for understanding the task:

<Few Shot Examples>

Now do the same for the following:
Plan: <Plan>
Chunks: <Candidate Chunk Set>

Figure 7: Prompt used by the Executor

Few Shot examples for Executor

Example 1

Plan:
1) Find the protagonist in the movie 'Inception'
2) Find the protagonist's birthplace
3) Find the population of that city

Chunks:
"Inception is a 2010 science fiction film starring Leonardo DiCaprio as Dom Cobb, a professional thief who infiltrates people's dreams..." "Leonardo DiCaprio was born in Los Angeles, California, United States on November 11, 1974..." "Los Angeles is the most populous city in California with a population of approximately 4 million people as of 2023..."

For each question in the plan, judge if the information needed is clearly and explicitly present in the chunks. The chunk set should collectively cover all different types of information required by the steps.

1. Score: 5. Short Explanation: Leonardo DiCaprio is explicitly named as the protagonist.
2. Score: 5. Short Explanation: Born in Los Angeles, California is directly stated.
3. Score: 5. Short Explanation: Population of 4 million is explicitly mentioned.
Total Score: 15

Example 2

Plan:
1) Find the director of the movie 'Titanic'
2) Find the director's net worth
3) Find the director's upcoming projects

Chunks:
"Titanic is a 1997 epic romance and disaster film. The movie was a massive box office success..." "The film won 11 Academy Awards including Best Picture and Best Director..." "Leonardo DiCaprio and Kate Winslet starred as the main characters in this epic love story..."

For each question in the plan, judge if the information needed is clearly and explicitly present in the chunks. The chunk set should collectively cover all different types of information required by the steps.

1. Score: 0. Short Explanation: Director name not mentioned, only that it won Best Director
2. Score: 0. Short Explanation: No financial information about anyone provided
3. Score: 0. Short Explanation: No information about future projects mentioned
Total Score: 0

Figure 8: Few Shot Examples used by the Executor

Few Shot Examples contd.

Example 3

Plan:
1) Find the author of 'Harry Potter'
2) Find the author's age
3) Find the number of books in the series

Chunks: "Harry potter, the book written by the sister of James Franco is fantasy book series that became globally popular..." " James Franco and along with all his siblings travelled to Europe..." "In Europe, Franco and JK Rowling played Hokkey like all the siblings" "The series consists of seven main novels plus several companion books..." "Rowling has become one of the most successful authors in modern publishing..."

For each question in the plan, judge if the information needed is clearly and explicitly present in the chunks. The chunk set should collectively cover all different types of information required by the steps.
1. Score: 3. Short Explanation: J.K. Rowling is inferred named as the author from the relationships.
2. Score: 0. Short Explanation: Age not mentioned anywhere in the chunks.
3. Score: 5. Short Explanation: Seven main novels is directly stated.
Total Score: 8

Figure 9: Few Shot Examples used by the Executor contd.

Generator

Instruction: You are a question answering assistant.

Task: Answer the question based on the given passages. Only give me the answer and do not output any other words. The following are given passages. <Context>
Answer the question based on the given passages. Only give me the answer and do not output any other words.
Question: <Query>
Answer:

Figure 10: Prompt used by the Generator

Dataset	DF-RAG + IC (uniform sampling)		DF-RAG + IC (Binary-Search)	
	Qwen 2.5 72B	Llama 3.3 70B	Qwen 2.5 72B	Llama 3.3 70B
HotPotQA	60.1	56.8	56.0	54.1
MuSiQue	32.7	34.0	30.3	33.4
2WikiMQA	56.8	55.2	54.3	50.5

Table 7: Comparison of DF-RAG + IC with uniform sampling and Binary-Search based sampling.