Efficient Context Selection for Long-Context QA: No Tuning, No Iteration, Just Adaptive-k

Anonymous ACL submission

Abstract

Retrieval-augmented generation (RAG) and long-context language models (LCLMs) both address context limitations of LLMs in open-domain question answering (QA). However, optimal external context to retrieve remains an open problem: fixing the retrieval size risks either wasting tokens or omitting key evidence. Existing adaptive methods like Self-RAG and SELF-ROUTE rely on iterative LLM prompting and perform well on factoid QA, but struggle with aggregation QA, where the optimal context size is both unknown and variable.

We present **Adaptive-k retrieval**, a simple and effective single-pass method that adaptively selects the number of passages based on the distribution of the similarity scores between the query and the candidate passages. It does not require model fine-tuning, extra LLM inferences or changes to existing retriever–reader pipelines. On both factoid and aggregation QA benchmarks, Adaptive-k matches or outperforms fixed-k baselines while using up to 10× fewer tokens than full-context input, yet still retrieves 70% of relevant passages. It improves accuracy across five LCLMs and two embedding models, highlighting that dynamically adjusting context size leads to more efficient and accurate QA.

1 Introduction

Despite remarkable progress in LLMs, efficiently incorporating external knowledge during inference for long or dynamic contexts remains a key challenge. Two major paradigms have emerged to address this: long-context language models (LCLMs), which extend the model's context window to directly ingest more information, and retrieval-augmented generation (RAG), which retrieves relevant documents from an external corpus to condition the generation. While these approaches are sometimes presented as alternatives (Li et al., 2024a; Yu et al., 2024), recent studies highlight their complementary nature (Li et al., 2024b).

A central bottleneck in both paradigms is determining how much context to include. Fixed-size retrieval budgets (*e.g.*, top-k retrieval) are suboptimal, because they either retrieve too little and risk omitting key evidence, or retrieve too much, which can overwhelm the model, increase latency and costs, and degrade performance (Yu et al., 2024; Leng et al., 2024; Jin et al., 2024). As Yang (2024) observes, the challenge in long-context reasoning lies not only in document length but also in how relevant information is distributed and duplicated within the context. Crucially, query type plays a major role: factoid questions may need only a few targeted facts, while aggregation queries (Maekawa et al., 2025) often require reasoning based on information from multiple evidence spans. This variability makes fixed-k retrieval suboptimal for complex tasks.

To address this, several hybrid and adaptive retrieval methods such as Self-RAG (Asai et al., 2023), Adaptive-RAG (Jeong et al., 2024), and Dynamic context cutoff (Xie et al., 2025) have been proposed, which estimate retrieval depth via iterative prompting, each time fetching a fixed number of documents. However, they assume white-box access to the LLM: Self-RAG requires fine-tuning the LLM, while dynamic context cutoff depends on access to internal KV cache states. This makes them incompatible with closed-source or API-based LLMs. While effective on factoid-style questions, they also face significant limitations in terms of scalability, latency, and deployment flexibility. Although SELF-ROUTE (Li et al., 2024b) offers a more modular solution, it still relies on a fixed retrieval size and lacks the ability to adapt to varying information needs across queries and context documents. This motivates our core research question: *How can we estimate the optimal number of passages to retrieve for a given query and set of* *context documents, without supervision or iterative prompting?*

To address this question, we introduce **Adaptive-k retrieval**, a simple yet effective plug-and-play method for dynamically selecting a query- and context-specific number of documents in a single retrieval pass. Our approach relies on analyzing the distribution of similarity scores between a query and candidate documents. By identifying the largest gap in the sorted similarity distribution, it estimates an optimal cutoff point, retrieving the top-k documents before the gap. Unlike prior adaptive retrieval methods, Adaptive-k requires no model fine-tuning, no access to internal components and no iterative prompting. It is fully modular, allowing seamless integration with existing retriever–reader pipeline and compatibility with black-box LLMs. By relying solely on the distributional structure of similarity scores, Adaptive-k adjusts the retrieval size on a per-query basis. This simple yet principled strategy leads to significant reductions in input length and inference cost, while maintaining or even improving the answer quality across both factoid and aggregation-style QA tasks. We compare Adaptive-k retrieval to prior approaches in Table 1.

We evaluate Adaptive-k on both factoid and aggregation-style QA tasks across multiple LCLMs and embedding models. Our experiments span two representative long-context benchmarks: HELMET (Yen et al., 2025), which includes factoid QA tasks with up to 128k-token contexts, and HoloBench (Maekawa et al., 2025), which focuses on aggregation-style queries. Our results show that on aggregation-QA, Adaptive-k outperforms SELF-ROUTE by up to +9 points in answer accuracy on high-information tasks. It consistently maintains $\sim$70% context recall and reduces token usage by 2× to 10× compared to full-context baselines. On factoid QA, Adaptive-k matches or exceeds the accuracy of fixed-size retrieval with up to 99% reduction in input tokens, effectively pruning irrelevant content. These findings highlight the importance of query-specific context sizing and establish Adaptive-k as a simple, robust, and efficient alternative to more complex adaptive retrieval strategies.

In summary, our key contributions are:

- We propose Adaptive-k, a simple yet effective plug-and-play method for adaptive document retrieval that dynamically adjusts context size based on similarity distribution statistics.
- Adaptive-k achieves higher accuracy than prior methods and up to 99% token reduction on factoid and aggregation QA against LCLMs with full context.
- We show that no single fixed-size retrieval strategy fits all settings. In contrast, Adaptive-k shows robust performance across multiple LLMs, embedding models and benchmarks.

2 Related Work

RAG and LCLMs are two prominent paradigms for equipping LLMs with external knowledge. Recent studies show that LCLMs can match or outperform RAG in certain QA tasks (Li et al., 2024a; Yu et al., 2024), yet the two methods are fundamentally complementary.

Several approaches have been proposed to leverage both the strengths of RAG and LCLMs with flexible retrieval strategies. Self-RAG (Asai et al., 2023) trains an LLM to generate reflection tokens that enable retrieval on the fly, so that the LLM can determine whether it needs any additional document by itself. SELF-ROUTE (Li et al., 2024b) asks an LLM whether it can answer the query with the retrieved context; if not, the LLM is given the full context. Adaptive-RAG (Jeong et al., 2024) uses a workflow that iteratively asks an LLM whether it can answer the given query with the retrieved context. LC-Boost (Qian et al., 2024) enables short-context LLMs to tackle long-context tasks by first identifying relevant information, then reasoning over it, without needing extended context windows or fine-tuning.

While effective in controlled settings, these methods often rely on white-box access to the LLM, fine-tuning, or multiple LLM inferences. Existing research has highlighted key limitations in RAG systems, particularly in terms of cost, modularity, and retrieval granularity. However, prior methods typically address these issues in isolation, and to our knowledge, no single approach has tackled all three challenges holistically. Our method is the first to offer a unified solution that is cost-efficient, modular, and capable of adaptive, query-specific retrieval in a single pass.

Cost. High-quality inference often comes with high token usage, energy consumption, and latency (Li et al., 2024b; Qian et al., 2024), underscoring the need for more cost-effective alternatives.

	Plug-and-Play via API	Retrieval Amount Variability	Single Retrieval Operation
No RAG (LCLM)	✓	✗	No Retrieval
RAG (traditional)	✓	✗	✓
Self-RAG (Asai et al., 2023)	✗	✓	✗
Adaptive-RAG (Jeong et al., 2024)	✓	✓	✗
SELF-ROUTE (Li et al., 2024b)	✓	✗	✓
LC-Boost (Qian et al., 2024)	✓	✓	✗
Dynamic context cutoff (Xie et al., 2025)	✗	✓	✗
Adaptive-k RAG (ours)	✓	✓	✓

Table 1: The comparison of previously proposed approaches as enhanced RAG. *Plug-and-Play via API* refers to whether the approach can be easily plugged in to various LLM pipelines. *Retrieval Amount Variability* refers to whether the system can flexibly change the retrieval amount depending on different queries and context. *Single Retrieval Operation* refers to whether the retrieval is performed in a single step or in multiple steps.

Modularity. Modularity is crucial for real-world deployment (Wang et al., 2024), but many existing methods require fine-tuning or training the LLM itself. This tight coupling reduces compatibility with API-based or closed-source models, limiting practical applicability.

Retrieval granularity. Aggregation-type queries often require comprehensive evidence and holistic understanding. For example, answering "Which colleges in California have over 10,000 students?" demands access to the full set of relevant entries. Fixed-size or iterative retrieval methods struggle with such cases, as they cannot dynamically adjust retrieval depth based on query complexity.

3 Method

This section details our approach to adaptive retrieval, grounded in the analysis of similarity score patterns to determine retrieval sizes adaptively based on the query and the context. We first review the standard RAG retrieval process, then present our methodology to identify the optimal threshold in similarity distributions to efficiently select relevant documents.

3.1 Retrieval in vanilla RAG

RAG consists of two steps: retrieval and generation. Given a query q and N context documents $C = \{c_i\}_1^N$, the retriever module identifies top-k semantically similar context documents C'. Modern RAG approaches convert the query and the context documents in natural language into the query embedding $\boldsymbol{q} \in \mathbb{R}^d$ and context embeddings $\boldsymbol{C} \in \mathbb{R}^{N \times d}$. Similarity scores $\boldsymbol{s} \in \mathbb{R}^N$ are then computed to quantify relevance, commonly using cosine similarity:

$$\boldsymbol{s} = f_{\text{sim}}(\boldsymbol{q}, \boldsymbol{C}) = \frac{\boldsymbol{C}\boldsymbol{q}^\top}{||\boldsymbol{q}|| \cdot ||\boldsymbol{C}||_{\text{rows}}}$$

RAG typically retrieves a fixed number of top-k documents (or tokens) based on the practitioner's choice. This fixed retrieval size is simple and modular but may result in inefficient token usage, either retrieving irrelevant documents or missing critical information, especially when the amount of relevant context varies depending on the provided context documents and the query type.

3.2 Toward efficient adaptive retrieval

Design motivation and principles. While vanilla RAG offers modularity and straightforward integration, its fixed retrieval size limits performance and efficiency in scenarios where the quantity of relevant context varies unpredictably such as in aggregation QA in the HoloBench benchmark (Maekawa et al., 2025). To address these limitations, we aim to design an adaptive retrieval mechanism that: (1) operates independently of the underlying inference model and requires no additional training or fine-tuning (**Plug-and-Play**), (2) flexibly controls the retrieval amount for each query, avoiding both wasting tokens and omitting key evidence (**Retrieval Amount Variability**), and (3) operates in a single pass without requiring iterative LLM calls (**Single Retrieval Operation**).

Preliminary analysis. To ground our design in empirical evidence, we conduct an in-depth analysis of the distributional patterns of cosine similarity scores between queries and candidate documents, which, crucially, are inference model-agnostic signals. This preliminary analysis reveals distinct distributional characteristics that inform our adaptive retrieval strategy.

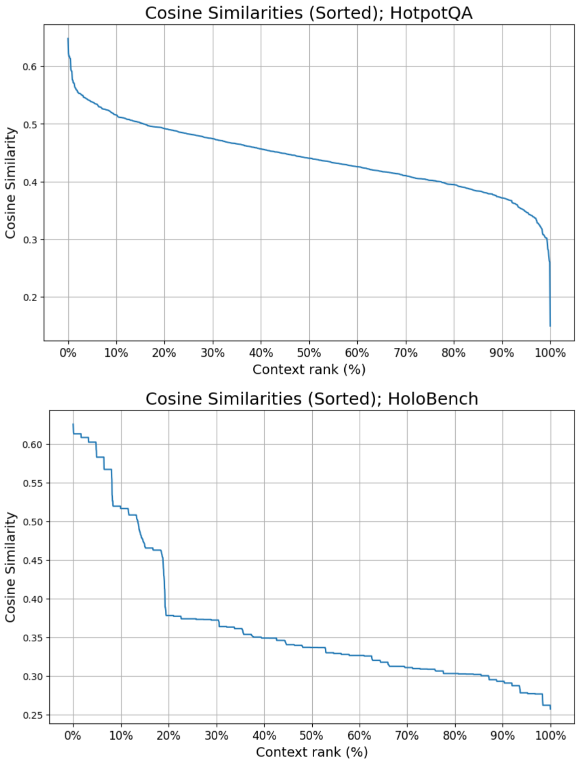

Figure 1: Example distributions of sorted cosine similarities from the long-context version of HotpotQA (Yang et al., 2018) included in HELMET (Yen et al., 2025) with 1,000 context documents (top) and HoloBench (Maekawa et al., 2025) with 10% relevant information amount (bottom). BAAI's bge-large-en-v1.5 is used as the embedding model.

As shown in Figure 1, for factoid QA tasks such as HotpotQA, the sorted similarity scores typically exhibit a pronounced gap separating a cluster of highly relevant documents from the rest, suggesting a natural threshold for retrieval. In contrast, aggregation tasks (e.g., HoloBench) show more irregular patterns, with gaps dispersed throughout the distribution – reflecting the variable spread of relevant information. In the bottom example in Figure 1, the 100k-token context is generated such that 10% of it is information relevant to the query. Indeed, the large gaps are observed around the top 5% to 20% context, aligning with our expectations.

These insights lead to the hypothesis that the largest gap in sorted similarity scores corresponds to the boundary between relevant and irrelevant documents, thus providing a data-driven criterion for adaptive retrieval size selection.

3.3 Proposed method

Building on these observations, we formalize an algorithm that adaptively estimates the retrieval

Algorithm 1 Adaptive k Estimation via Largest Similarity Gap

Require: q, C, `Embedder`$(\cdot)$, `Similarity`$(\cdot)$
Ensure: Estimated k such that the largest similarity drop occurs before the k-th item

$\boldsymbol{q} \leftarrow$ `Embedder`(q)
$\boldsymbol{C} \leftarrow$ `Embedder`(C) ▷ Precomputed
$\boldsymbol{s} \leftarrow$ `Similarity`$(\boldsymbol{q}, \boldsymbol{C})$
Sort $\boldsymbol{s}$ in descending order
$\boldsymbol{g} \leftarrow$ `array`$()$ ▷ For storing the gap
for $i = 0$ to $|\boldsymbol{s}| - 2$ **do**
  Append $\boldsymbol{s}[i] - \boldsymbol{s}[i+1]$ to $\boldsymbol{g}$
end for
$k \leftarrow \arg\max(\boldsymbol{g})$ ▷ Index at the largest gap
return k

threshold k by identifying the position of the steepest drop in the similarity score distribution. The method proceeds as follows: Compute the cosine similarities $\boldsymbol{s}$ of the query $\boldsymbol{q}$ and context documents $\boldsymbol{C}$. Sort the scores in descending order. Compute their first discrete differences $\boldsymbol{g}$ and choose the index k where the similarity drop is the largest. Figure 2 depicts this process within the RAG workflow. Under the assumption that the embeddings of documents are precomputed, the time complexity of this algorithm is $\mathcal{O}(n \log n)$. The algorithm is described in Algorithm 1.

In practice, while determining the threshold k based on the largest similarity gap is effective, a naïve implementation might miss relevant documents located immediately beyond the identified threshold. To address this, we incorporate a small fixed buffer, retrieving an additional B documents after the k-th document. In our experiments, we set $B = 5$. Furthermore, as depicted in Figure 1, the largest gap may occasionally manifest among the least relevant documents, leading to the retrieval of an excessively large portion of the context. To avoid this and align with our focus on retrieval from extremely long contexts, we restrict the search for the largest gap to the top 90% of documents sorted by their similarity scores.

4 Experimental setup

In our experiments, we aim to answer the following research questions:

- How does the proposed adaptive-k method compare to other modular retrieval approaches on aggregation tasks with varying amounts of

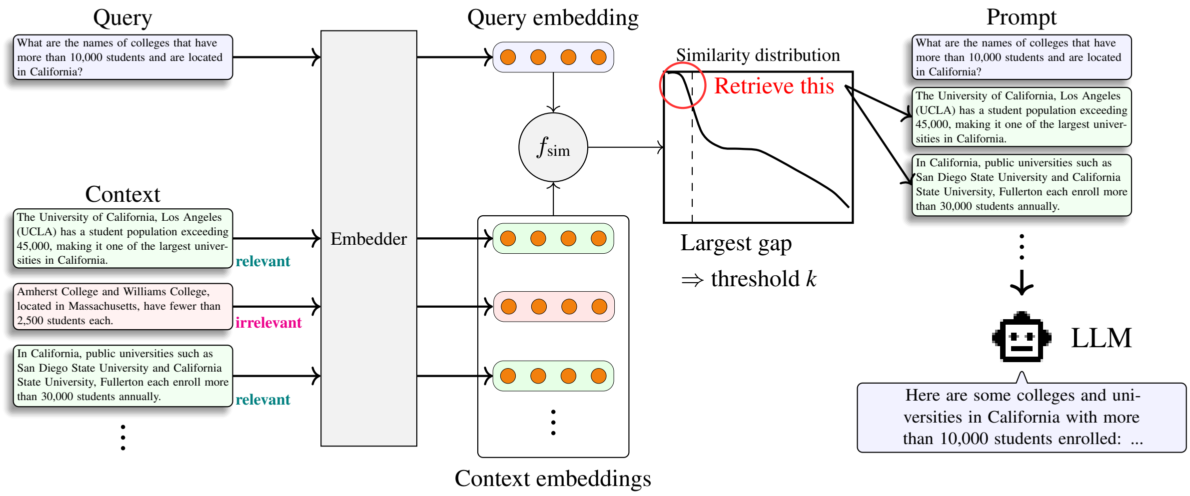

Figure 2: The proposed method in the RAG workflow. The method chooses the threshold k for retrieval based on a large gap in the sorted similarity score distribution.

relevant context?

- How does performance of Adaptive-k vary across factoid QA and aggregation QA tasks?
- How does the performance gain from Adaptive-k retrieval vary across LLMs?
- How do different embedding models influence the performance of Adaptive-k?

To answer these questions, we employ the experimental settings detailed below.

4.1 Dataset

For testing on factoid QA tasks, we use HotpotQA (Yang et al., 2018), Natural Questions (NQ) (Kwiatkowski et al., 2019), and TriviaQA (Joshi et al., 2017), as curated by HELMET (Yen et al., 2025) for long-context benchmarking with 128k input tokens. Due to the high computational cost of long-context inference, we evaluate on a subset of 100 examples per dataset.

For aggregation tasks, we employ HoloBench (Maekawa et al., 2025), which provides 90 evaluation samples. HoloBench allows control over both total context size and the amount of information relevant to the query. We fix the total context to 100k tokens and evaluate under varying levels of relevant information, with `info_amount` = {5000, 10000, 25000, 50000} tokens.

4.2 Models

Retriever. We test our method on small and large embedding models: BAAI's bge-en-large-v1.5[1] (Xiao et al., 2023) and Alibaba NLP's gte-Qwen2-1.5B-instruct[2] (Li et al., 2023).

Reader. We use five closed and open models: GPT-4o-mini, GPT-4o (OpenAI et al., 2024), Gemini-2.5-Flash (Team et al., 2024), Llama4-Scout, and Llama4-Maverick (Touvron et al., 2023). The model details are provided in Appendix A.2.

4.3 Compared methods

We compare the proposed adaptive-k method against zero-shot LLMs (without context), LLMs with full context, and SELF-ROUTE (Li et al., 2024b), which is another modular retrieval method with a single retrieval step. In SELF-ROUTE, fixed top 5k tokens are retrieved for the first inference step. We also show the results of the fixed-n retrieval method with varying numbers of tokens n as performance references. Specifically, we run experiments with $n \in \{1000, 5000, 10000, 25000, 50000\}$ and regard the best-performing setting as the oracle. In this way, we can compare the performance of adaptive-k against the best possible score of the fixed retrieval method.

4.4 Metrics

To evaluate the retrieval performance, context recall (Ru et al., 2024) is computed, which represents how

[1] `https://huggingface.co/BAAI/bge-large-en-v1.5`

[2] `https://huggingface.co/Alibaba-NLP/gte-Qwen2-1.5B-instruct`

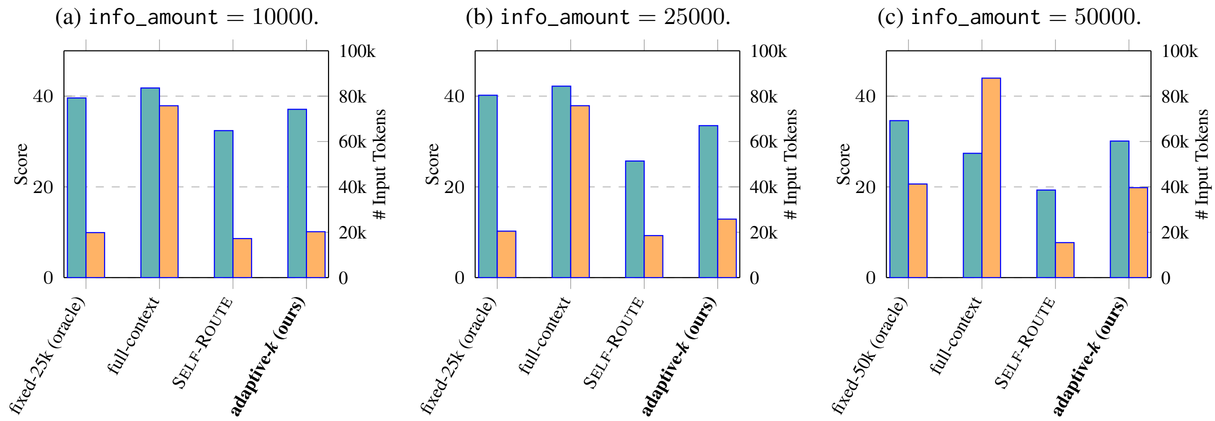

Figure 3: The results with different amounts of relevant information in the HoloBench tasks. The best-performing fixed-n setting is chosen as the oracle. ■ is for performance improvement, and ■ for the number of input tokens.

much of the relevant context documents were able to be retrieved. For the evaluation of generation performance, we use substring exact match (SubEM) for HotpotQA, NQ, and TriviaQA, and LLM-as-a-judge for HoloBench, following the metrics used in their original implementation in HELMET (Yen et al., 2025) and HoloBench, respectively. LLM-as-a-judge evaluates whether the generated answer contains a correct mention of the gold answer, assigning a score of 1 if it finds a correct mention, 0.5 for a partially correct mention, and 0 otherwise. For the judge model, GPT-4o-mini is used. To evaluate the inference cost, we count the number of input and output tokens, assuming that the financial cost on the user's end and energy consumption depends on the amount of tokens (Husom et al., 2024).

5 Results

This section provides the results of the experiments with a focus on different task types, reader models, and embedding models. For the full results, see Appendix A.3.

5.1 Aggregation-type QA

Figure 3 shows GPT-4o's results in the HoloBench tasks where each task is designed to contain different amounts of relevant information (`info_amount`: 10k, 25k, 50k tokens) in the context. It can be observed that our Adaptive-k method constantly outperforms SELF-ROUTE. The performance improvements of Adaptive-k are particularly notable when the amount of relevant information in the context is high. Also, our method flexibly increases the amount of retrieved context chunks when there is a higher amount of relevant information in the entire context. In contrast, SELF-ROUTE tends to underestimate the amount of relevant context and jump to a conclusion that the LLM can answer the query with the 5k-token context retrieved in the first round, leading to lower performance in a high amount of relevant information.

	info5k	info10k	info25k	info50k
SELF-ROUTE	65.79	45.04	30.42	21.54
Adaptive-k	**75.74**	**68.54**	**66.16**	**67.43**
fixed-1k	12.05	6.53	2.77	1.47
fixed-5k	51.92	31.77	14.06	7.54
fixed-10k	66.68	59.10	28.80	15.39
fixed-25k	78.48	78.18	68.13	39.55
fixed-50k	86.79	87.34	86.88	76.90

Table 2: A comparison of the context recall scores across different relevant information amounts in the HoloBench tasks. The query and contexts are embedded by bge-large-en-v1.5. The scores compared are SELF-ROUTE and Adaptive-k, as well as the results of fixed-n token retrieval as references.

This contrast is also reflected in the context recall scores. As shown in Table 2, Adaptive-k consistently achieves a context recall score of approximately 70 across varying levels of relevant information, indicating that it retrieves approximately 70% of truly relevant chunks regardless of their proportion in the full context. The contrast is even more pronounced when compared to context recall of SELF-ROUTE, with Adaptive-k achieving more than three times higher context recall.

5.2 Factoid-type QA

Figure 4 shows the comparison of Adaptive-k against the zero-shot setting, fixed 1k-token retrieval, full context, and SELF-ROUTE. All meth-

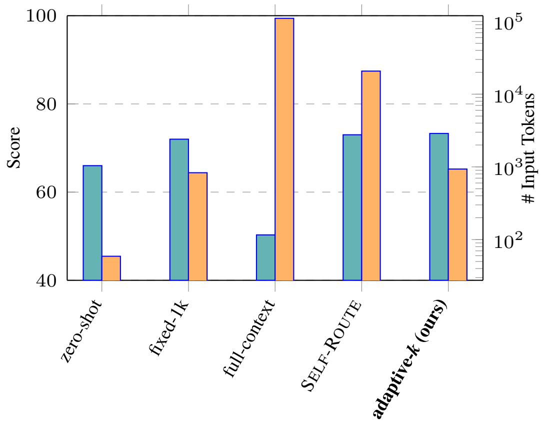

Figure 4: A performance comparison of our proposed method (**Adaptive-k**) in the factoid QA tasks against existing methods. The embedding model is bge-large-en-v1.5, and the reader model is GPT-4o. ▮ is for the SubEM scores, and ▮ for the number of input tokens.

	Retrieval	BGE	GTE
HotpotQA	SELF-ROUTE	90.83	25.83
	adaptive-k	70.83	5.50
NQ	SELF-ROUTE	51.90	20.67
	adaptive-k	27.20	2.85
TriviaQA	SELF-ROUTE	46.52	10.20
	adaptive-k	31.21	3.00
HoloBench-5k	SELF-ROUTE	65.79	65.18
	adaptive-k	75.74	82.20
HoloBench-10k	SELF-ROUTE	45.04	45.87
	adaptive-k	68.54	78.99
HoloBench-25k	SELF-ROUTE	30.42	31.02
	adaptive-k	66.16	76.47
HoloBench-50k	SELF-ROUTE	21.54	21.90
	adaptive-k	67.43	72.54

Table 3: A comparison of the context recall scores across tasks between BGE (bge-large-en-v1.5) and GTE (gte-Qwen2-1.5B-instruct).

ods are implemented using GPT-4o. Our method achieves a 99% reduction in input cost compared to the full context input, and a 90% reduction compared to SELF-ROUTE. Since users generally lack prior knowledge of the optimal retrieval size, Adaptive-k successfully reduces the cost while improving the generation quality compared to zero-shot question answering.

5.3 Comparison across LLMs

Since our methods only modify the retriever module, the retrieved documents to be fed into an LLM's prompt remain the same across different LLMs. However, we observe that its effectiveness varies notably by model. Figure 5 shows the average score improvements and input token counts across different relevant information settings in HoloBench. Larger high-performance LLMs such as GPT-4o (Figure 5b), Gemini-2.5-Flash (Figure 5c), and Llama4-Maverick (Figure 5e) show substantial gains from Adaptive-k retrieval compared to SELF-ROUTE. In contrast, smaller LLMs such as GPT-4o-mini (Figure 5a) and Llama4-Scout (Figure 5d) exhibit more modest improvements. Nonetheless, even for smaller models, Adaptive-k effectively reduces context length while maintaining performance close to the full-context and oracle fixed-n baselines.

5.4 Embedding bottleneck

We observed that the effectiveness of our adaptive method is sensitive to the choice of embedding model. As shown in Table 3, bge-large-en-v1.5 embeddings and gte-Qwen2-1.5B-instruct embeddings have different strengths depending on the task. In factoid QA tasks, BGE embeddings consistently yield higher context recall than GTE, whereas GTE performs better on HoloBench. The underlying cause remains unclear, but we identify a few potential factors: (1) Context chunk length: the factoid QA tasks in HELMET generally have a longer context chunk length (up to ∼100 tokens) than HoloBench (∼40 tokens); (2) Chunking scheme (Zhong et al., 2025): while the context chunks in HoloBench contain well-formed natural-language sentences, those in the factoid QA tasks often contain mid-sentence breaks; (3) Training scheme: differences in pretraining corpora and formatting may lead to divergent performance across embedding models. Overall, choosing the right embedding model is critical for ensuring RAG effectiveness. For general use, we recommend bge-large-en-v1.5 for Adaptive-k due to its strong and consistent performance across settings.

5.5 Limitation of fixed retrieval

While fixed-n retrieval occasionally outperforms Adaptive-k method, it requires prior knowledge of the optimal n, which is difficult to estimate in practice. Our results show that the best-performing n varies across task types, query types, embedding models, reader models, and the distribution of relevant information. In contrast, Adaptive-k is able to dynamically adjust the retrieval amount based on the query and context chunks, eliminating the need for manual tuning. This not only removes the

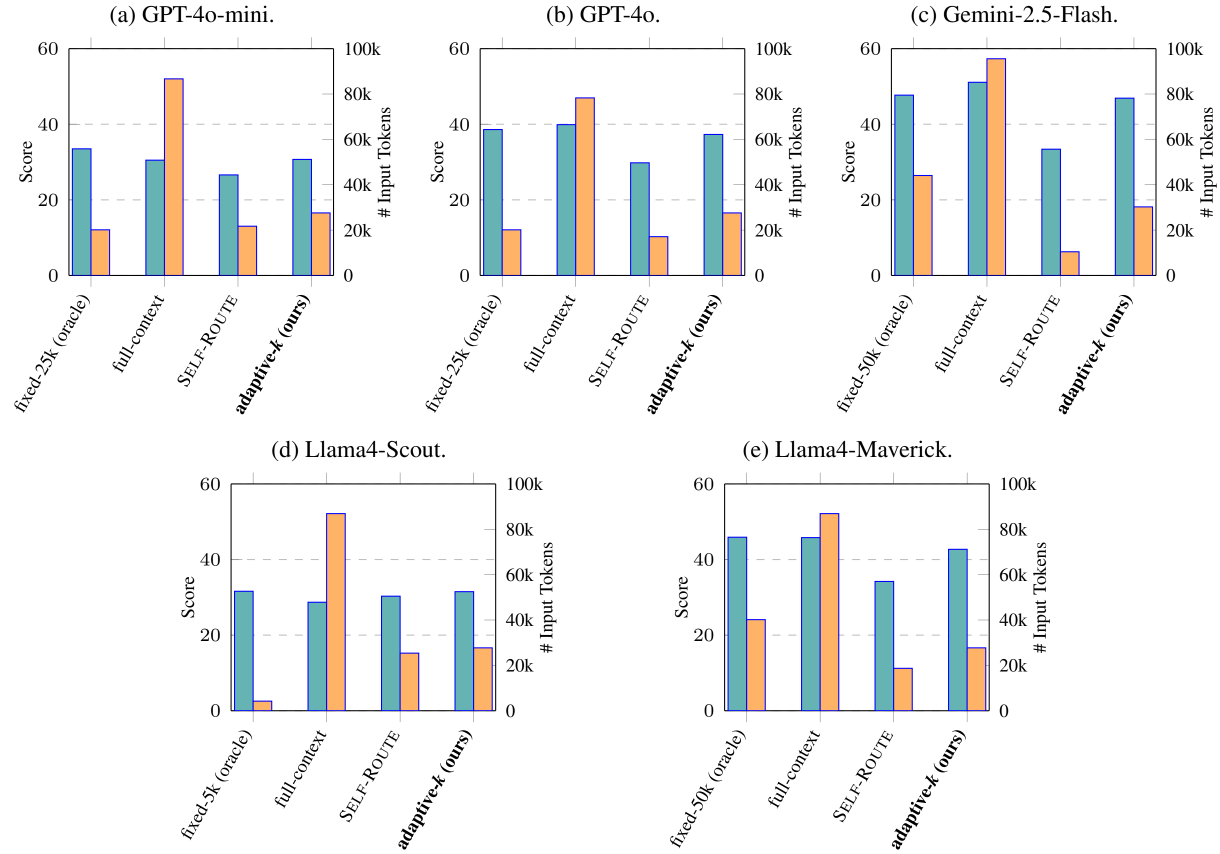

Figure 5: A performance comparison across the different reader models in the HoloBench task. The emnbedding model is bge-large-en-v1.5. ▮ is for performance improvement, and ▮ for the number of input tokens.

burden and risk of heuristically selecting an n but also provides a more robust and generalizable solution across a wide range of scenarios, especially in cases where the relevant context size is highly variable or unknown a priori.

6 Conclusion

We presented a simple yet effective and efficient plug-and-play method, **adaptive-k**, that dynamically selects the number of context chunks to retrieve in a single step, based on the similarity distribution between the query and context chunks. Unlike existing adaptive retrieval methods that requires iterative inference steps, our method only requires a single matrix calculation to estimate the retrieval threshold, achieving a fast and flexible retrieval module. This method is particularly effective for aggregation-type QA tasks, where the optimal number of context chunks varies across examples and cannot be predetermined by a fixed-token retrieval strategy. Results on HoloBench demonstrate that Adaptive-k flexibly adjusts retrieval size to align with the amount of relevant information in the context. In factoid QA tasks, where relevant information is sparse, our method aggressively prunes the context while still outperforming zero-shot QA in answer quality. Compared to SELF-ROUTE, our method consistently achieves superior performance in aggregation-type QA tasks, while drastically reducing the input size and maintaining higher context recall.

Our adaptive-k retrieval is a plug-and-play, single-pass alternative to fixed-size retrieval, yet several directions remain. First, because the method is orthogonal to most RAG pipelines, pairing it with techniques such as query-expansion, iterative reranking, or generative feedback loops could further improve accuracy and latency. Second, embedding models excel on different query and corpus traits; a runtime system that selects—or ensembles—embeddings per query may unlock extra gains in recall and robustness.

Limitations

While our proposed method shows promising results in adaptive retrieval for question answering tasks, it has several limitations that warrant discussion.

First, the method is not directly applicable to tasks such as summarization, where the objective is to process the entire input holistically rather than retrieve a subset of relevant context. In such cases, aggressive filtering may omit important information that contributes to the overall summary. In addition, an embedding model is not able to identify the relevant context documents with a general summarization-type query. For instance, when the query for a summarization task is a general statement like "`The summary of this book is:`" (an example from ∞BENCH Sum (Zhang et al., 2024)), the high-similarity context chunks do not necessarily reflect the importance to the answer because the query does not quite contain semantically significant information.

Second, our method is designed for natural language inputs and assumes meaningful semantic similarity between queries and context chunks. It does not generalize well to non-natural-language tasks, such as those involving structured key-value formats (*e.g.*, JSON), where semantic embeddings may not capture relevance effectively.

Third, the approach is sensitive to surface-level variations in text. For example, typographical errors in the query or context can negatively affect embedding quality and distort similarity scores, leading to suboptimal retrieval decisions. If the queries are expected to be noisy with non-standard spellings or grammar, adding a query standardization module (Chan et al., 2024) on top of our adaptive-k method would be helpful.

Lastly, the method may be vulnerable to adversarial or malicious inputs (Wallace et al., 2019). A specially crafted context chunk could receive an artificially high or low similarity score, thereby introducing a large gap in the similarity distribution and misleading the algorithm into selecting an incorrect retrieval threshold (Su et al., 2024). Mitigating such risks would require additional robustness checks or adversarial training techniques, which are beyond the scope of this work.

Ethical considerations

One of the key advantages of our proposed adaptive retrieval method is its potential to reduce the environmental impact of LLM inference. By discarding irrelevant context chunks and only retrieving a minimal yet sufficient subset of documents, our approach significantly reduces the number of input tokens processed. In our experiments, our proposed method discarded nearly 99% of the input tokens in factoid QA tasks, and substantially reduced input size in aggregation QA tasks while maintaining high context recall.

This reduction translates into lower computational overhead, leading to more energy-efficient inference. As a result, our method contributes to decreasing the carbon footprint associated with deploying LLMs at scale. With the growing trend of longer context windows, flexibly filtering out irrelevant context is necessary to ensure energy-efficient inference.

While efficiency is a central goal, we emphasize that any optimization must not compromise fairness or content coverage. Our method is designed to be model-agnostic and does not introduce or amplify biases beyond those present in the similarity scoring mechanism, *e.g.*, cosine similarity over embedding spaces. However, care should be taken when applying this method in high-stakes domains, *e.g.*, medical or legal QA, where discarding seemingly low-similarity context could result in the omission of critical information. Further research is needed to quantify such risks and guide responsible deployment.

While we used AI assitants such as ChatGPT and Copilot to assist in coding and revising this paper, we carefully reviewed and edited all content to ensure it meets our standards and aligns with our research goals.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *Preprint*, arXiv:2310.11511.

Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. 2024. Rq-rag: Learning to refine queries for retrieval augmented generation. *Preprint*, arXiv:2404.00610.

Erik Johannes Husom, Arda Goknil, Lwin Khin Shar, and Sagar Sen. 2024. The price of prompting: Profiling energy use in large language models inference. *Preprint*, arXiv:2407.16893.

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

Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O. Arik. 2024. Long-context llms meet rag: Overcoming challenges for long inputs in rag. *Preprint*, arXiv:2410.05983.

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.

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

Quinn Leng, Jacob Portes, Sam Havens, Matei Zaharia, and Michael Carbin. 2024. Long context rag performance of large language models. *Preprint*, arXiv:2411.03538.

Xinze Li, Yixin Cao, Yubo Ma, and Aixin Sun. 2024a. Long context vs. rag for llms: An evaluation and revisits. *Preprint*, arXiv:2501.01880.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024b. 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.

Seiji Maekawa, Hayate Iso, and Nikita Bhutani. 2025. Holistic reasoning with long-context lms: A benchmark for database operations on massive textual data. *Preprint*, arXiv:2410.11996.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Yujia Zhou, Xu Chen, and Zhicheng Dou. 2024. Are long-llms a necessity for long-context tasks? *Preprint*, arXiv:2405.15318.

Dongyu Ru, Lin Qiu, Xiangkun Hu, Tianhang Zhang, Peng Shi, Shuaichen Chang, Cheng Jiayang, Cunxiang Wang, Shichao Sun, Huanyu Li, Zizhao Zhang, Binjie Wang, Jiarong Jiang, Tong He, Zhiguo Wang, Pengfei Liu, Yue Zhang, and Zheng Zhang. 2024. Ragchecker: A fine-grained framework for diagnosing retrieval-augmented generation. *Preprint*, arXiv:2408.08067.

Jinyan Su, Jin Peng Zhou, Zhengxin Zhang, Preslav Nakov, and Claire Cardie. 2024. Towards more robust retrieval-augmented generation: Evaluating rag under adversarial poisoning attacks. *Preprint*, arXiv:2412.16708.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, and 1331 others. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, Ruicheng Yin, Changze Lv, Xiaoqing Zheng, and Xuanjing Huang. 2024. Searching for best practices in retrieval-augmented generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17716–17736, Miami, Florida, USA. Association for Computational Linguistics.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

Roy Xie, Junlin Wang, Paul Rosu, Chunyuan Deng, Bolun Sun, Zihao Lin, and Bhuwan Dhingra. 2025. Knowing when to stop: Dynamic context cutoff for large language models. *Preprint*, arXiv:2502.01025.

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.

Zi Yang. 2024. Retrieval or holistic understanding? dolce: Differentiate our long context evaluation tasks. *Preprint*, arXiv:2409.06338.

Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. 2025. Helmet: How to evaluate long-context language models effectively and thoroughly. *Preprint*, arXiv:2410.02694.

Tan Yu, Anbang Xu, and Rama Akkiraju. 2024. In defense of rag in the era of long-context language models. *Preprint*, arXiv:2409.01666.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024. ∞bench: Extending long context evaluation beyond 100k tokens. *Preprint*, arXiv:2402.13718.

Zijie Zhong, Hanwen Liu, Xiaoya Cui, Xiaofan Zhang, and Zengchang Qin. 2025. Mix-of-granularity: Optimize the chunking granularity for retrieval-augmented generation. *Preprint*, arXiv:2406.00456.

A Appendix

A.1 Prompt templates

A.1.1 Prompt template for the factoid QA tasks

````
Your task is to answer the question provided. To help you answer accurately, some relevant
context documents have been retrieved. After reviewing them, you'll be asked the same question
again. Please respond succinctly.

**Input:**
- **Question:**
```
{question}
```

- **Context:**
```
{context}
```

- **Question:**
```
{question}
```


**Response:**
- **Answer:**
````

A.1.2 Prompt template for the HoloBench tasks

````
You'll be given a set of sentences to read through carefully. Once you've reviewed them, I'll
ask you a question related to the information in those sentences. Your job is to think
critically about the details, analyze the sentences in relation to the question, and then
provide your answer. If the information clearly supports a partial answer, provide that.
However, if the evidence is unclear or insufficient, it is okay to respond with "No answer."

**Input:**
- **Sentences:**
```
{context}
```

- **Question:**
```
{question}
```

**Response:**
- **Reasoning:**
  - [Describe how you thought through the sentences and how they helped you reach your
    conclusion. If the evidence is unclear or insufficient to provide a reliable answer,
    explain why. Your reasoning should not exceed 10,000 words.]
- **Answer:** [Provide an answer only if it is clearly supported by the information in the
    sentences. If the evidence is unclear or insufficient, respond with "No answer."]
````

A.1.3 Prompt template for LLM-as-a-Judge

````
You will be given a question along with a response generated by an assistant and the
corresponding ground truth data. Your task is to assess the response based on its accuracy
and completeness in comparison to the ground truth. For each entry in the ground truth,
determine whether the information provided by the assistant is an "Exact Match," a "Partial
Match," or a "No Match."

#### **Evaluation Criteria:**

- **Exact Match**: The assistant's response precisely matches the ground truth in both
    content and detail.
- **Partial Match**: The assistant's response includes some correct information but is either
    incomplete, incorrectly ordered, or contains inaccuracies.
- **No Match**: The assistant's response does not accurately reflect the ground truth or is
    missing entirely.

#### **Special Cases:**

**Ground Truth is None**:
- If the ground truth is `None` (represented as an empty list `[]`):
  - **Exact Match**: If the assistant's response indicates that there is no information or
    content.
  - **No Match**: If the assistant's response provides any information when the ground truth
    is `None`.

#### **Output Format:**

- The output should be a list of objects where each object contains:
  - An `"id"` that matches the `id` of the corresponding ground truth entry.
  - A `"label"` indicating whether the assistant's response is an `"Exact Match"`, `"Partial
    Match"`, or `"No Match"`.

- The number of output objects should match the number of entries in the ground truth.

---

### **Examples:**
{in_context_examples}

====== Your task starts here ======

**Question:**
```
{question}
```

**Assistant's Response:**
```
{pred}
```

**Ground Truth:**
```
{gold}
```

**Output Format:**
```
{output_format}
```
````

A.2 Detailed experimental setup

We set temperature and top-p parameters to 0.0 and 1.0, respectively, for all our experiments. For Gemini-2.5-Flash, we set its thinking budget to 0. Table 4 lists the models used in our experiments.

Model	**Size**	**Context**	**Model name / snapshot**	**License**
GPT-4o	—	128k	`gpt-4o-2024-08-06`	OpenAI Service Terms[3]
GPT-4o-mini	—	128k	`gpt-4o-mini-2024-07-18`	OpenAI Service Terms
Gemini-2.5-Flash	—	1M	`gemini-2.5-flash-preview-04-17`	Gemini API Additional Terms of Service[4]
Llama-4-Maverick	400B	1M	`meta-llama/Llama-4-Maverick-17B-128E-Instruct`	Llama 4 Community License Agreement[5]
Llama-4-Scout	109B	10M	`meta-llama/Llama-4-Scout-17B-16E-Instruct`	Llama 4 Community License Agreement

Table 4: A list of the LLMs used in the experiments. An em-dash (—) means that the model size is not publicly disclosed.

[3] https://openai.com/policies/services-agreement/ [Accessed: May 12, 2025]
[4] https://ai.google.dev/gemini-api/terms [Accessed: May 12, 2025]
[5] https://www.llama.com/llama4/license/ [Accessed: May 12, 2025]

A.3 Full results

A.3.1 Factoid QA tasks (BGE embeddings)

Task	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
HotpotQA	zeroshot	39	0.00 ± 0.00	100.00 ± 0.00	63.03 ± 6.98	15.56 ± 9.92
	fixed-1k	60	69.33 ± 30.77	99.33 ± 0.09	852.30 ± 57.33	20.53 ± 13.61
	fixed-5k	66	84.50 ± 26.93	96.46 ± 0.29	3983.81 ± 219.82	20.87 ± 14.45
	fixed-10k	66	88.50 ± 23.05	92.89 ± 0.55	7911.32 ± 440.28	21.85 ± 14.35
	fixed-25k	67	92.50 ± 20.15	82.19 ± 1.32	19763.59 ± 1109.32	22.36 ± 14.48
	fixed-50k	66	95.33 ± 14.23	64.46 ± 2.45	39597.66 ± 2224.65	24.00 ± 15.87
	full-context	45	100.00 ± 0.00	0.00 ± 0.00	109666.92 ± 5537.59	16.49 ± 18.35
	SELF-ROUTE	61	90.83 ± 22.77	75.25 ± 40.17	28008.57 ± 45573.55	17.31 ± 17.84
	adaptive-*k*	**63**	70.83 ± 31.01	99.24 ± 0.17	954.33 ± 206.78	20.46 ± 14.43
NQ	zeroshot	49	0.00 ± 0.00	100.00 ± 0.00	53.37 ± 2.29	23.36 ± 15.22
	fixed-1k	54	26.45 ± 30.50	99.36 ± 0.09	806.77 ± 70.12	28.50 ± 18.53
	fixed-5k	59	42.45 ± 36.68	96.66 ± 0.28	3837.48 ± 333.10	31.33 ± 21.80
	fixed-10k	58	50.35 ± 35.94	93.27 ± 0.52	7632.00 ± 655.62	32.11 ± 23.94
	fixed-25k	62	62.78 ± 32.87	83.10 ± 1.25	19051.28 ± 1574.22	33.91 ± 26.90
	fixed-50k	59	68.89 ± 29.96	66.16 ± 2.45	38102.93 ± 2989.91	37.39 ± 29.33
	full-context	41	100.00 ± 0.00	0.00 ± 0.00	110607.54 ± 4711.16	23.10 ± 27.28
	SELF-ROUTE	**55**	52.72 ± 36.37	77.33 ± 38.86	25839.53 ± 44249.67	23.60 ± 20.09
	adaptive-*k*	54	27.20 ± 31.58	99.25 ± 0.25	927.69 ± 283.52	28.83 ± 18.99
TriviaQA	zeroshot	83	0.00 ± 0.00	100.00 ± 0.00	60.09 ± 7.79	7.85 ± 6.27
	fixed-1k	94	31.21 ± 36.58	99.34 ± 0.10	833.75 ± 67.42	11.86 ± 9.07
	fixed-5k	93	42.10 ± 39.93	96.53 ± 0.26	3913.01 ± 260.44	11.73 ± 9.41
	fixed-10k	92	49.90 ± 40.11	93.03 ± 0.49	7772.18 ± 488.60	12.64 ± 11.19
	fixed-25k	93	54.74 ± 40.26	82.59 ± 1.14	19384.51 ± 1164.94	13.64 ± 11.10
	fixed-50k	94	61.66 ± 37.83	65.21 ± 2.27	38819.58 ± 2326.33	15.72 ± 11.95
	full-context	61	100.00 ± 0.00	0.00 ± 0.00	110733.69 ± 3419.97	11.95 ± 13.89
	SELF-ROUTE	90	48.19 ± 39.84	84.95 ± 31.53	17112.00 ± 35900.97	8.69 ± 8.41
	adaptive-*k*	**92**	31.21 ± 36.58	99.26 ± 0.23	918.86 ± 240.86	11.69 ± 9.15
Average	zeroshot	57.00	0.00	0.00	58.83	15.59
	fixed-1k	69.33	42.33	99.34	830.94	20.30
	fixed-5k	72.67	56.35	96.55	3911.43	21.31
	fixed-10k	72.00	62.92	93.07	7771.83	22.20
	fixed-25k	74.00	70.00	82.63	19399.79	23.30
	fixed-50k	73.00	75.29	65.28	38840.06	25.70
	full-context	49.00	100.00	0.00	110336.05	17.18
	SELF-ROUTE	68.67	63.91	79.18	23653.37	16.53
	adaptive-*k*	**69.67**	43.08	99.25	933.63	20.33

Table 5: Full GPT-4o-mini's results in the factoid QA tasks.

Task	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
HotpotQA	zeroshot	50	0.00 ± 0.00	100.00 ± 0.00	63.03 ± 6.98	19.31 ± 16.53
	fixed-1k	61	69.33 ± 30.77	99.33 ± 0.09	852.30 ± 57.33	27.72 ± 18.35
	fixed-5k	70	84.50 ± 26.93	96.46 ± 0.29	3983.81 ± 219.82	27.25 ± 17.25
	fixed-10k	76	88.50 ± 23.05	92.89 ± 0.55	7911.32 ± 440.28	29.16 ± 18.31
	fixed-25k	74	92.50 ± 20.15	82.19 ± 1.32	19763.59 ± 1109.32	28.66 ± 20.86
	fixed-50k	73	95.33 ± 14.23	64.46 ± 2.45	39597.66 ± 2224.65	27.89 ± 20.22
	full-context	48	100.00 ± 0.00	0.00 ± 0.00	109666.92 ± 5537.59	18.91 ± 20.30
	SELF-ROUTE	**66**	84.50 ± 26.93	96.46 ± 0.29	23663.11 ± 42241.38	22.49 ± 20.04
	adaptive-*k*	63	70.83 ± 31.01	99.24 ± 0.17	954.33 ± 206.78	28.24 ± 19.56
NQ	zeroshot	57	0.00 ± 0.00	100.00 ± 0.00	53.37 ± 2.29	27.38 ± 24.23
	fixed-1k	59	26.45 ± 30.50	99.36 ± 0.09	806.77 ± 70.12	33.62 ± 26.13
	fixed-5k	64	42.45 ± 36.68	96.66 ± 0.28	3837.48 ± 333.10	37.37 ± 30.24
	fixed-10k	64	50.35 ± 35.94	93.27 ± 0.52	7632.00 ± 655.62	38.41 ± 32.61
	fixed-25k	64	62.78 ± 32.87	83.10 ± 1.25	19051.28 ± 1574.22	38.48 ± 33.35
	fixed-50k	63	68.89 ± 29.96	66.16 ± 2.45	38102.93 ± 2989.91	39.76 ± 33.99
	full-context	41	100.00 ± 0.00	0.00 ± 0.00	110607.54 ± 4711.16	24.74 ± 33.79
	SELF-ROUTE	**61**	42.45 ± 36.68	96.66 ± 0.28	24713.66 ± 43308.30	33.64 ± 34.15
	adaptive-*k*	**61**	27.20 ± 31.58	99.25 ± 0.25	927.69 ± 283.52	34.85 ± 27.40
TriviaQA	zeroshot	91	0.00 ± 0.00	100.00 ± 0.00	60.09 ± 7.79	8.33 ± 7.16
	fixed-1k	96	31.21 ± 36.58	99.34 ± 0.10	833.75 ± 67.42	16.10 ± 11.89
	fixed-5k	95	42.10 ± 39.93	96.53 ± 0.26	3913.01 ± 260.44	16.01 ± 11.49
	fixed-10k	94	49.90 ± 40.11	93.03 ± 0.49	7772.18 ± 488.60	15.55 ± 9.91
	fixed-25k	93	54.74 ± 40.26	82.59 ± 1.14	19384.51 ± 1164.94	15.96 ± 9.81
	fixed-50k	93	61.66 ± 37.83	65.21 ± 2.27	38819.58 ± 2326.33	16.31 ± 10.62
	full-context	62	100.00 ± 0.00	0.00 ± 0.00	110733.69 ± 3419.97	12.94 ± 12.48
	SELF-ROUTE	92	42.10 ± 39.93	96.53 ± 0.26	13900.19 ± 31870.44	11.39 ± 10.31
	adaptive-*k*	**96**	31.21 ± 36.58	99.26 ± 0.23	918.86 ± 240.86	16.10 ± 12.07
Average	zeroshot	66.00	0.00	0.00	58.83	18.34
	fixed-1k	72.00	42.33	99.34	830.94	25.81
	fixed-5k	76.33	56.35	96.55	3911.43	26.88
	fixed-10k	78.00	62.92	93.07	7771.83	27.71
	fixed-25k	77.00	70.00	82.63	19399.79	27.70
	fixed-50k	76.33	75.29	65.28	38840.06	27.99
	full-context	50.33	100.00	0.00	110336.05	18.86
	SELF-ROUTE	73.00	56.35	96.55	20758.99	22.51
	adaptive-*k*	**73.33**	43.08	99.25	933.63	26.40

Table 6: Full GPT-4o's results in the factoid QA tasks.

Task	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
HotpotQA	zeroshot	46	0.00 ± 0.00	100.00 ± 0.00	57.25 ± 7.72	3.00 ± 1.62
	fixed-1k	54	69.33 ± 30.77	99.33 ± 0.09	883.55 ± 67.88	15.24 ± 23.19
	fixed-5k	63	84.50 ± 26.93	96.46 ± 0.29	4170.89 ± 274.70	12.92 ± 19.22
	fixed-10k	66	88.50 ± 23.05	92.89 ± 0.55	8295.62 ± 548.91	12.62 ± 17.17
	fixed-25k	66	92.50 ± 20.15	82.19 ± 1.32	20727.06 ± 1379.61	15.09 ± 20.48
	fixed-50k	72	95.33 ± 14.23	64.46 ± 2.45	41530.89 ± 2780.34	16.97 ± 22.12
	full-context	71	100.00 ± 0.00	0.00 ± 0.00	115121.31 ± 5964.79	17.60 ± 19.32
	SELF-ROUTE	**68**	95.33 ± 17.42	69.45 ± 43.53	36820.47 ± 52663.73	13.95 ± 19.00
	adaptive-*k*	55	70.83 ± 31.01	99.24 ± 0.17	990.43 ± 216.72	15.29 ± 24.47
NQ	zeroshot	47	0.00 ± 0.00	100.00 ± 0.00	46.30 ± 2.46	4.67 ± 3.92
	fixed-1k	44	26.45 ± 30.50	99.36 ± 0.09	826.53 ± 80.39	26.14 ± 33.25
	fixed-5k	59	42.45 ± 36.68	96.66 ± 0.28	3959.77 ± 351.93	31.82 ± 32.77
	fixed-10k	59	50.35 ± 35.94	93.27 ± 0.52	7891.38 ± 704.11	115.35 ± 816.66
	fixed-25k	62	62.78 ± 32.87	83.10 ± 1.25	19730.89 ± 1697.34	34.87 ± 53.13
	fixed-50k	61	68.89 ± 29.96	66.16 ± 2.45	39505.22 ± 3266.78	35.23 ± 47.76
	full-context	64	100.00 ± 0.00	0.00 ± 0.00	115142.91 ± 5115.01	28.15 ± 22.01
	SELF-ROUTE	**60**	54.71 ± 36.63	74.41 ± 40.87	30547.34 ± 48902.74	27.40 ± 30.96
	adaptive-*k*	47	27.20 ± 31.58	99.25 ± 0.25	951.38 ± 295.47	29.04 ± 32.84
TriviaQA	zeroshot	93	0.00 ± 0.00	100.00 ± 0.00	54.36 ± 8.36	2.55 ± 1.48
	fixed-1k	87	31.21 ± 36.58	99.34 ± 0.10	859.44 ± 78.27	8.99 ± 16.82
	fixed-5k	93	42.10 ± 39.93	96.53 ± 0.26	4073.06 ± 317.10	8.74 ± 15.25
	fixed-10k	92	49.90 ± 40.11	93.03 ± 0.49	8097.22 ± 589.74	9.30 ± 14.94
	fixed-25k	93	54.74 ± 40.26	82.59 ± 1.14	20213.64 ± 1424.47	8.41 ± 11.68
	fixed-50k	92	61.66 ± 37.83	65.21 ± 2.27	40490.49 ± 2852.03	11.95 ± 14.79
	full-context	95	100.00 ± 0.00	0.00 ± 0.00	115669.41 ± 4229.98	10.48 ± 8.73
	SELF-ROUTE	**94**	49.19 ± 39.24	84.95 ± 31.53	17948.59 ± 37786.95	7.39 ± 10.84
	adaptive-*k*	86	31.21 ± 36.58	99.26 ± 0.23	947.50 ± 244.93	7.53 ± 13.74
Average	zeroshot	62.00	0.00	0.00	52.64	3.41
	fixed-1k	61.67	42.33	99.34	856.51	16.79
	fixed-5k	71.67	56.35	96.55	4067.91	17.83
	fixed-10k	72.33	62.92	93.07	8094.74	45.76
	fixed-25k	73.67	70.00	82.63	20223.86	19.46
	fixed-50k	75.00	75.29	65.28	40508.87	21.38
	full-context	76.67	100.00	0.00	115311.21	18.74
	SELF-ROUTE	**74.00**	66.41	76.27	28438.80	16.25
	adaptive-*k*	62.67	43.08	99.25	963.10	17.29

Table 7: Full Gemini-2.5-Flash's results in the factoid QA tasks.

Task	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
HotpotQA	zeroshot	38	0.00 ± 0.00	100.00 ± 0.00	61.17 ± 7.05	226.87 ± 1635.13
	fixed-1k	65	69.33 ± 30.77	99.33 ± 0.09	850.33 ± 59.50	59.78 ± 97.50
	fixed-5k	65	84.50 ± 26.93	96.46 ± 0.29	4006.32 ± 227.71	41.23 ± 63.66
	fixed-10k	68	88.50 ± 23.05	92.89 ± 0.55	7963.66 ± 456.87	38.49 ± 60.86
	fixed-25k	68	92.50 ± 20.15	82.19 ± 1.32	19912.13 ± 1152.36	33.14 ± 50.95
	fixed-50k	67	95.33 ± 14.23	64.46 ± 2.45	39898.83 ± 2310.69	36.25 ± 56.47
	full-context	67	100.00 ± 0.00	0.00 ± 0.00	110457.05 ± 5390.30	36.77 ± 65.52
	SELF-ROUTE	**73**	89.50 ± 23.53	84.89 ± 31.51	17292.33 ± 36140.60	60.39 ± 78.22
	adaptive-*k*	63	70.83 ± 31.01	99.24 ± 0.17	952.64 ± 206.19	53.92 ± 87.49
NQ	zeroshot	58	0.00 ± 0.00	100.00 ± 0.00	51.54 ± 2.41	62.84 ± 53.57
	fixed-1k	62	26.45 ± 30.50	99.36 ± 0.09	802.11 ± 71.62	60.35 ± 48.38
	fixed-5k	66	42.45 ± 36.68	96.66 ± 0.28	3847.40 ± 341.79	73.75 ± 67.45
	fixed-10k	64	50.35 ± 35.94	93.27 ± 0.52	7660.11 ± 675.79	76.13 ± 61.56
	fixed-25k	66	62.78 ± 32.87	83.10 ± 1.25	19136.92 ± 1631.03	85.91 ± 81.62
	fixed-50k	68	68.89 ± 29.96	66.16 ± 2.45	38284.28 ± 3103.43	288.08 ± 1664.19
	full-context	68	100.00 ± 0.00	0.00 ± 0.00	111154.46 ± 4416.57	113.88 ± 202.07
	SELF-ROUTE	**66**	48.21 ± 36.70	85.06 ± 31.57	17228.44 ± 36372.48	142.10 ± 316.76
	adaptive-*k*	61	27.20 ± 31.58	99.25 ± 0.25	923.58 ± 286.51	67.89 ± 59.55
TriviaQA	zeroshot	85	0.00 ± 0.00	100.00 ± 0.00	58.29 ± 7.89	20.12 ± 39.32
	fixed-1k	98	31.21 ± 36.58	99.34 ± 0.10	830.54 ± 66.35	24.97 ± 54.24
	fixed-5k	98	42.10 ± 39.93	96.53 ± 0.26	3928.14 ± 261.91	24.80 ± 63.48
	fixed-10k	97	49.90 ± 40.11	93.03 ± 0.49	7808.89 ± 495.48	18.29 ± 42.28
	fixed-25k	96	54.74 ± 40.26	82.59 ± 1.14	19486.58 ± 1189.90	19.47 ± 51.48
	fixed-50k	95	61.66 ± 37.83	65.21 ± 2.27	39032.69 ± 2399.86	12.60 ± 22.71
	full-context	96	100.00 ± 0.00	0.00 ± 0.00	111322.11 ± 3373.09	23.69 ± 52.77
	SELF-ROUTE	97	47.69 ± 40.13	89.78 ± 24.76	11742.38 ± 28544.35	34.38 ± 63.98
	adaptive-*k*	**98**	31.21 ± 36.58	99.26 ± 0.23	915.89 ± 238.98	30.32 ± 73.20
Average	zeroshot	60.33	0.00	0.00	57.00	103.28
	fixed-1k	75.00	42.33	99.34	827.66	48.37
	fixed-5k	76.33	56.35	96.55	3927.29	46.59
	fixed-10k	76.33	62.92	93.07	7810.89	44.30
	fixed-25k	76.67	70.00	82.63	19511.88	46.17
	fixed-50k	76.67	75.29	65.28	39071.93	112.31
	full-context	77.00	100.00	0.00	110977.87	58.11
	SELF-ROUTE	**78.67**	61.80	86.57	15421.05	78.96
	adaptive-*k*	74.00	43.08	99.25	930.70	50.71

Table 8: Full Llama4-Scout's results in the factoid QA tasks.

Task	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
HotpotQA	zeroshot	52	0.00 ± 0.00	100.00 ± 0.00	61.17 ± 7.05	99.43 ± 110.61
	fixed-1k	71	69.33 ± 30.77	99.33 ± 0.09	850.33 ± 59.50	110.04 ± 141.75
	fixed-5k	78	84.50 ± 26.93	96.46 ± 0.29	4006.32 ± 227.71	68.99 ± 88.75
	fixed-10k	74	88.50 ± 23.05	92.89 ± 0.55	7963.66 ± 456.87	44.12 ± 63.11
	fixed-25k	71	92.50 ± 20.15	82.19 ± 1.32	19912.13 ± 1152.36	43.30 ± 55.48
	fixed-50k	72	95.33 ± 14.23	64.46 ± 2.45	39898.83 ± 2310.69	45.51 ± 66.82
	full-context	75	100.00 ± 0.00	0.00 ± 0.00	110457.05 ± 5390.30	41.18 ± 54.88
	SELF-ROUTE	**79**	86.00 ± 26.35	92.61 ± 19.00	8383.42 ± 21450.47	90.55 ± 120.15
	adaptive-*k*	71	70.83 ± 31.01	99.24 ± 0.17	952.64 ± 206.19	112.62 ± 146.43
NQ	zeroshot	53	0.00 ± 0.00	100.00 ± 0.00	51.54 ± 2.41	54.22 ± 54.55
	fixed-1k	63	26.45 ± 30.50	99.36 ± 0.09	802.11 ± 71.62	68.16 ± 58.41
	fixed-5k	65	42.45 ± 36.68	96.66 ± 0.28	3847.40 ± 341.79	75.36 ± 69.75
	fixed-10k	67	50.35 ± 35.94	93.27 ± 0.52	7660.11 ± 675.79	75.90 ± 79.63
	fixed-25k	64	62.78 ± 32.87	83.10 ± 1.25	19136.92 ± 1631.03	69.41 ± 69.85
	fixed-50k	64	68.89 ± 29.96	66.16 ± 2.45	38284.28 ± 3103.43	71.23 ± 77.03
	full-context	67	100.00 ± 0.00	0.00 ± 0.00	111154.46 ± 4416.57	66.25 ± 66.57
	SELF-ROUTE	**65**	45.95 ± 37.23	90.86 ± 23.07	10536.57 ± 26530.78	69.87 ± 75.65
	adaptive-*k*	62	27.20 ± 31.58	99.25 ± 0.25	923.58 ± 286.51	75.56 ± 73.03
TriviaQA	zeroshot	91	0.00 ± 0.00	100.00 ± 0.00	58.29 ± 7.89	25.52 ± 54.17
	fixed-1k	96	31.21 ± 36.58	99.34 ± 0.10	830.54 ± 66.35	39.52 ± 57.18
	fixed-5k	98	42.10 ± 39.93	96.53 ± 0.26	3928.14 ± 261.91	31.02 ± 48.63
	fixed-10k	98	49.90 ± 40.11	93.03 ± 0.49	7808.89 ± 495.48	21.72 ± 32.93
	fixed-25k	96	54.74 ± 40.26	82.59 ± 1.14	19486.58 ± 1189.90	22.79 ± 46.88
	fixed-50k	96	61.66 ± 37.83	65.21 ± 2.27	39032.69 ± 2399.86	15.46 ± 34.71
	full-context	96	100.00 ± 0.00	0.00 ± 0.00	111322.11 ± 3373.09	19.58 ± 34.15
	SELF-ROUTE	**98**	44.60 ± 40.03	93.64 ± 16.55	7440.28 ± 19973.57	37.87 ± 63.00
	adaptive-*k*	96	31.21 ± 36.58	99.26 ± 0.23	915.89 ± 238.98	49.73 ± 70.11
Average	zeroshot	65.33	0.00	0.00	57.00	59.72
	fixed-1k	76.67	42.33	99.34	827.66	72.57
	fixed-5k	80.33	56.35	96.55	3927.29	58.46
	fixed-10k	79.67	62.92	93.07	7810.89	47.25
	fixed-25k	77.00	70.00	82.63	19511.88	45.17
	fixed-50k	77.33	75.29	65.28	39071.93	44.07
	full-context	79.33	100.00	0.00	110977.87	42.34
	SELF-ROUTE	**80.67**	58.85	92.37	8786.76	66.10
	adaptive-*k*	76.33	43.08	99.25	930.70	79.30

Table 9: Full Llama4-Maverick's results in the factoid QA tasks.

A.3.2 HoloBench (BGE embeddings)

Info amount	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
info5k	zeroshot	10.00	0.00 ± 0.00	100.00 ± 0.00	58.13	43.34
	fixed-1k	28.19	12.05 ± 6.42	99.18 ± 0.43	1000.07	325.14
	fixed-5k	38.74	51.92 ± 29.82	95.85 ± 1.86	4194.01	923.67
	fixed-10k	43.50	66.68 ± 30.73	91.80 ± 2.92	8011.64	801.81
	fixed-25k	39.81	78.48 ± 26.96	79.59 ± 4.56	19574.94	1489.67
	fixed-50k	37.67	86.79 ± 20.88	57.76 ± 5.92	39224.80	2342.40
	full-context	37.76	100.00 ± 0.00	0.00 ± 0.00	85882.50	3147.27
	SELF-ROUTE	31.32	69.46 ± 27.62	74.68 ± 40.18	23044.97	1523.99
	adaptive-*k*	**40.86**	75.74 ± 30.48	74.07 ± 25.68	24625.02	2220.94
info10k	zeroshot	6.22	0.00 ± 0.00	100.00 ± 0.00	58.13	45.28
	fixed-1k	22.55	6.53 ± 3.11	99.19 ± 0.40	1003.83	322.87
	fixed-5k	34.06	31.77 ± 14.86	95.84 ± 1.90	4228.71	987.09
	fixed-10k	34.85	59.10 ± 27.14	91.74 ± 3.47	8235.02	1712.06
	fixed-25k	36.44	78.18 ± 26.55	79.55 ± 5.42	19838.92	1716.79
	fixed-50k	29.98	87.34 ± 20.51	57.88 ± 6.41	39437.42	2910.81
	full-context	26.59	100.00 ± 0.00	0.00 ± 0.00	86139.74	4370.90
	SELF-ROUTE	28.56	48.18 ± 27.16	77.76 ± 37.78	19627.93	2228.59
	adaptive-*k*	**33.16**	68.54 ± 32.55	79.22 ± 21.59	20233.99	2338.73
info25k	zeroshot	4.22	0.00 ± 0.00	100.00 ± 0.00	58.13	43.17
	fixed-1k	16.67	2.77 ± 1.15	99.21 ± 0.32	999.62	331.77
	fixed-5k	25.76	14.06 ± 5.50	95.96 ± 1.56	4215.72	670.87
	fixed-10k	28.61	28.80 ± 9.87	91.86 ± 3.09	8269.06	1554.73
	fixed-25k	32.63	68.13 ± 22.77	79.60 ± 7.10	20475.06	2249.00
	fixed-50k	30.89	86.88 ± 20.14	58.46 ± 8.58	40152.73	3203.13
	full-context	29.53	100.00 ± 0.00	0.00 ± 0.00	86751.63	3767.70
	SELF-ROUTE	**27.01**	34.19 ± 35.59	74.68 ± 40.17	22726.84	1090.31
	adaptive-*k*	25.68	66.16 ± 36.90	73.86 ± 23.04	25778.38	2818.23
info50k	zeroshot	5.28	0.00 ± 0.00	100.00 ± 0.00	58.13	43.59
	fixed-1k	11.73	1.47 ± 0.55	99.22 ± 0.23	1006.91	310.46
	fixed-5k	20.88	7.54 ± 2.52	96.02 ± 1.12	4235.36	511.71
	fixed-10k	21.53	15.39 ± 4.25	92.01 ± 2.13	8288.68	1556.82
	fixed-25k	25.29	39.55 ± 8.89	79.75 ± 5.38	20622.82	2609.31
	fixed-50k	30.18	76.90 ± 16.82	58.85 ± 9.99	41264.63	2885.44
	full-context	27.98	100.00 ± 0.00	0.00 ± 0.00	87936.41	3051.19
	SELF-ROUTE	19.57	26.38 ± 37.06	76.86 ± 38.66	21444.74	945.72
	adaptive-*k*	**22.93**	67.43 ± 38.13	60.73 ± 25.94	39654.20	2781.81
Average	zeroshot	6.43	0.00	0.00	58.13	43.84
	fixed-1k	19.78	5.70	99.20	1002.61	322.56
	fixed-5k	29.86	26.32	95.92	4218.45	773.33
	fixed-10k	32.12	42.49	91.85	8201.10	1406.36
	fixed-25k	33.54	66.09	79.62	20127.94	2016.19
	fixed-50k	32.18	84.48	58.24	40019.90	2835.45
	full-context	30.47	100.00	0.00	86677.57	3584.26
	SELF-ROUTE	26.61	44.55	76.00	21711.12	1447.15
	adaptive-*k*	**30.66**	69.47	71.97	27572.90	2539.93

Table 10: Full GPT-4o-mini's results in the HoloBench tasks.

Info amount	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
info5k	zeroshot	7.22	0.00 ± 0.00	100.00 ± 0.00	58.13	65.21
	fixed-1k	26.14	12.05 ± 6.42	99.18 ± 0.43	1000.07	289.39
	fixed-5k	42.32	51.92 ± 29.82	95.85 ± 1.86	4194.01	913.32
	fixed-10k	49.79	66.68 ± 30.73	91.80 ± 2.92	8011.64	1334.67
	fixed-25k	46.27	78.48 ± 26.96	79.59 ± 4.56	19574.94	2087.61
	fixed-50k	43.82	86.79 ± 20.88	57.76 ± 5.92	39224.80	3188.69
	full-context	48.30	100.00 ± 0.00	0.00 ± 0.00	73652.50	3680.13
	SELF-ROUTE	41.86	65.79 ± 27.76	76.70 ± 38.60	17260.98	1139.29
	adaptive-*k*	**48.60**	75.74 ± 30.48	74.07 ± 25.68	24625.02	1362.71
info10k	zeroshot	5.11	0.00 ± 0.00	100.00 ± 0.00	58.13	64.00
	fixed-1k	21.83	6.53 ± 3.11	99.19 ± 0.40	1003.83	298.53
	fixed-5k	32.45	31.77 ± 14.86	95.84 ± 1.90	4228.71	1001.83
	fixed-10k	36.48	59.10 ± 27.14	91.74 ± 3.47	8235.02	2589.58
	fixed-25k	39.65	78.18 ± 26.55	79.55 ± 5.42	19838.92	3527.68
	fixed-50k	38.55	87.34 ± 20.51	57.88 ± 6.41	39437.42	4061.82
	full-context	41.75	100.00 ± 0.00	0.00 ± 0.00	75768.00	4767.44
	SELF-ROUTE	32.37	45.04 ± 25.18	79.96 ± 36.00	17208.18	1146.12
	adaptive-*k*	**37.06**	68.54 ± 32.55	79.22 ± 21.59	20233.99	2252.20
info25k	zeroshot	3.54	0.00 ± 0.00	100.00 ± 0.00	58.13	65.69
	fixed-1k	16.51	2.77 ± 1.15	99.21 ± 0.32	999.62	282.18
	fixed-5k	27.52	14.06 ± 5.50	95.96 ± 1.56	4215.72	1350.80
	fixed-10k	29.10	28.80 ± 9.87	91.86 ± 3.09	8269.06	2520.64
	fixed-25k	40.25	68.13 ± 22.77	79.60 ± 7.10	20475.06	3406.40
	fixed-50k	34.18	86.88 ± 20.14	58.46 ± 8.58	40152.73	4366.80
	full-context	42.24	100.00 ± 0.00	0.00 ± 0.00	75787.37	4802.50
	SELF-ROUTE	25.71	30.42 ± 32.70	77.87 ± 37.82	18525.29	1759.81
	adaptive-*k*	**33.46**	66.16 ± 36.90	73.86 ± 23.04	25778.38	4017.11
info50k	zeroshot	5.19	0.00 ± 0.00	100.00 ± 0.00	58.13	62.30
	fixed-1k	11.59	1.47 ± 0.55	99.22 ± 0.23	1006.91	274.89
	fixed-5k	20.62	7.54 ± 2.52	96.02 ± 1.12	4235.36	735.80
	fixed-10k	23.15	15.39 ± 4.25	92.01 ± 2.13	8288.68	1595.42
	fixed-25k	28.11	39.55 ± 8.89	79.75 ± 5.38	20622.82	4269.89
	fixed-50k	34.58	76.90 ± 16.82	58.85 ± 9.99	41264.63	4244.67
	full-context	27.40	100.00 ± 0.00	0.00 ± 0.00	87936.41	3051.19
	SELF-ROUTE	19.28	21.54 ± 31.97	80.09 ± 36.03	15426.00	917.82
	adaptive-*k*	**30.10**	67.43 ± 38.13	60.73 ± 25.94	39654.20	4374.22
Average	zeroshot	5.26	0.00	0.00	58.13	64.30
	fixed-1k	19.02	5.70	99.20	1002.61	286.25
	fixed-5k	30.73	26.32	95.92	4218.45	1000.44
	fixed-10k	34.63	42.49	91.85	8201.10	2010.08
	fixed-25k	38.57	66.09	79.62	20127.94	3322.89
	fixed-50k	37.78	84.48	58.24	40019.90	3965.49
	full-context	39.92	100.00	0.00	78286.07	4075.32
	SELF-ROUTE	29.80	40.70	78.65	17105.11	1240.76
	adaptive-*k*	**37.30**	69.47	71.97	27572.90	3001.56

Table 11: Full GPT-4o's results in the HoloBench tasks.

Info amount	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
info5k	zeroshot	10.19	0.00 ± 0.00	100.00 ± 0.00	52.64	644.08
	fixed-1k	27.78	12.05 ± 6.42	99.18 ± 0.43	1091.11	478.93
	fixed-5k	49.11	51.92 ± 29.82	95.85 ± 1.86	4610.40	1470.04
	fixed-10k	54.52	66.68 ± 30.73	91.80 ± 2.92	8760.92	2743.83
	fixed-25k	55.31	78.48 ± 26.96	79.59 ± 4.56	21300.94	3422.06
	fixed-50k	56.37	86.79 ± 20.88	57.76 ± 5.92	42764.79	3995.68
	full-context	63.27	100.00 ± 0.00	0.00 ± 0.00	94227.37	4584.58
	SELF-ROUTE	47.56	57.27 ± 31.23	88.38 ± 25.87	11559.12	1894.97
	adaptive-*k*	**55.68**	75.74 ± 30.48	74.07 ± 25.68	26762.53	2776.24
info10k	zeroshot	8.33	0.00 ± 0.00	100.00 ± 0.00	52.64	572.52
	fixed-1k	21.09	6.53 ± 3.11	99.19 ± 0.40	1099.17	469.12
	fixed-5k	34.94	31.77 ± 14.86	95.84 ± 1.90	4683.90	2107.79
	fixed-10k	50.51	59.10 ± 27.14	91.74 ± 3.47	9105.06	2855.68
	fixed-25k	55.24	78.18 ± 26.55	79.55 ± 5.42	21717.67	4696.43
	fixed-50k	54.06	87.34 ± 20.51	57.88 ± 6.41	43101.23	6040.41
	full-context	53.65	100.00 ± 0.00	0.00 ± 0.00	94626.73	6381.93
	SELF-ROUTE	35.72	36.72 ± 22.19	89.40 ± 24.10	10196.77	2031.94
	adaptive-*k*	**56.26**	68.54 ± 32.55	79.22 ± 21.59	22081.99	3637.80
info25k	zeroshot	6.28	0.00 ± 0.00	100.00 ± 0.00	52.64	657.34
	fixed-1k	15.42	2.77 ± 1.15	99.21 ± 0.32	1098.33	455.64
	fixed-5k	31.18	14.06 ± 5.50	95.96 ± 1.56	4699.37	1695.57
	fixed-10k	37.86	28.80 ± 9.87	91.86 ± 3.09	9230.02	3053.66
	fixed-25k	42.87	68.13 ± 22.77	79.60 ± 7.10	22790.66	6461.89
	fixed-50k	44.54	86.88 ± 20.14	58.46 ± 8.58	44317.83	7982.89
	full-context	42.19	100.00 ± 0.00	0.00 ± 0.00	95716.53	9289.30
	SELF-ROUTE	28.12	20.06 ± 22.11	89.55 ± 24.11	11120.20	2086.64
	adaptive-*k*	**43.76**	66.16 ± 36.90	73.86 ± 23.04	28207.51	5901.29
info50k	zeroshot	6.95	0.00 ± 0.00	100.00 ± 0.00	52.64	735.01
	fixed-1k	9.77	1.47 ± 0.55	99.22 ± 0.23	1108.12	445.62
	fixed-5k	22.66	7.54 ± 2.52	96.02 ± 1.12	4730.58	1836.44
	fixed-10k	30.00	15.39 ± 4.25	92.01 ± 2.13	9277.87	2990.94
	fixed-25k	33.71	39.55 ± 8.89	79.75 ± 5.38	23092.23	6596.67
	fixed-50k	35.68	76.90 ± 16.82	58.85 ± 9.99	46101.70	8373.53
	full-context	45.44	100.00 ± 0.00	0.00 ± 0.00	97597.56	8792.94
	SELF-ROUTE	22.37	11.75 ± 19.29	91.76 ± 19.93	8923.41	1610.90
	adaptive-*k*	**31.72**	67.43 ± 38.13	60.73 ± 25.94	43888.98	7643.86
Average	zeroshot	7.94	0.00	0.00	52.64	652.24
	fixed-1k	18.51	5.70	99.20	1099.18	462.33
	fixed-5k	34.47	26.32	95.92	4681.06	1777.46
	fixed-10k	43.22	42.49	91.85	9093.47	2911.03
	fixed-25k	46.78	66.09	79.62	22225.38	5294.26
	fixed-50k	47.66	84.48	58.24	44071.39	6598.13
	full-context	51.14	100.00	0.00	95542.05	7262.19
	SELF-ROUTE	33.44	31.45	89.78	10449.88	1906.11
	adaptive-*k*	**46.85**	69.47	71.97	30235.25	4989.80

Table 12: Full Gemini-2.5-Flash's results in the HoloBench tasks.

Info amount	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
info5k	zeroshot	9.49	0.00 ± 0.00	100.00 ± 0.00	56.18	266.46
	fixed-1k	29.52	12.05 ± 6.42	99.18 ± 0.43	994.79	450.14
	fixed-5k	40.38	51.92 ± 29.82	95.85 ± 1.86	4195.84	650.72
	fixed-10k	40.73	66.68 ± 30.73	91.80 ± 2.92	8024.09	687.39
	fixed-25k	37.33	78.48 ± 26.96	79.59 ± 4.56	19625.13	711.01
	fixed-50k	34.47	86.79 ± 20.88	57.76 ± 5.92	39342.62	707.11
	full-context	36.32	100.00 ± 0.00	0.00 ± 0.00	86093.44	789.47
	SELF-ROUTE	36.35	70.18 ± 29.18	69.25 ± 43.22	27700.34	785.69
	adaptive-*k*	**39.01**	75.74 ± 30.48	74.07 ± 25.68	24711.58	1170.32
info10k	zeroshot	7.01	0.00 ± 0.00	100.00 ± 0.00	56.18	260.24
	fixed-1k	19.25	6.53 ± 3.11	99.19 ± 0.40	999.90	473.22
	fixed-5k	33.58	31.77 ± 14.86	95.84 ± 1.90	4233.97	708.81
	fixed-10k	31.87	59.10 ± 27.14	91.74 ± 3.47	8254.73	796.22
	fixed-25k	29.75	78.18 ± 26.55	79.55 ± 5.42	19898.58	840.93
	fixed-50k	28.60	87.34 ± 20.51	57.88 ± 6.41	39561.99	1070.07
	full-context	30.50	100.00 ± 0.00	0.00 ± 0.00	86347.34	1143.99
	SELF-ROUTE	33.51	51.87 ± 29.27	72.57 ± 41.54	25307.50	1167.84
	adaptive-*k*	**34.69**	68.54 ± 32.55	79.22 ± 21.59	20348.23	709.23
info25k	zeroshot	7.23	0.00 ± 0.00	100.00 ± 0.00	56.18	277.41
	fixed-1k	17.62	2.77 ± 1.15	99.21 ± 0.32	997.53	473.57
	fixed-5k	29.44	14.06 ± 5.50	95.96 ± 1.56	4226.03	682.94
	fixed-10k	28.95	28.80 ± 9.87	91.86 ± 3.09	8298.52	759.61
	fixed-25k	31.39	68.13 ± 22.77	79.60 ± 7.10	20562.19	795.73
	fixed-50k	28.35	86.88 ± 20.14	58.46 ± 8.58	40309.13	1268.03
	full-context	25.94	100.00 ± 0.00	0.00 ± 0.00	86997.00	961.38
	SELF-ROUTE	**29.08**	32.25 ± 34.30	76.81 ± 38.65	21041.81	914.91
	adaptive-*k*	26.90	66.16 ± 36.90	73.86 ± 23.04	25958.96	854.74
info50k	zeroshot	6.18	0.00 ± 0.00	100.00 ± 0.00	56.18	427.23
	fixed-1k	11.93	1.47 ± 0.55	99.22 ± 0.23	1004.93	416.12
	fixed-5k	22.89	7.54 ± 2.52	96.02 ± 1.12	4246.40	639.41
	fixed-10k	23.75	15.39 ± 4.25	92.01 ± 2.13	8317.29	735.40
	fixed-25k	23.94	39.55 ± 8.89	79.75 ± 5.38	20706.06	956.62
	fixed-50k	25.46	76.90 ± 16.82	58.85 ± 9.99	41427.09	854.82
	full-context	22.10	100.00 ± 0.00	0.00 ± 0.00	88197.90	1321.16
	SELF-ROUTE	22.24	32.58 ± 40.92	70.50 ± 42.76	27508.79	980.10
	adaptive-*k*	**25.45**	67.43 ± 38.13	60.73 ± 25.94	39897.64	1182.12
Average	zeroshot	7.48	0.00	0.00	56.18	307.84
	fixed-1k	19.58	5.70	99.20	999.29	453.26
	fixed-5k	31.57	26.32	95.92	4225.56	670.47
	fixed-10k	31.33	42.49	91.85	8223.66	744.66
	fixed-25k	30.60	66.09	79.62	20197.99	826.07
	fixed-50k	29.22	84.48	58.24	40160.21	975.01
	full-context	28.72	100.00	0.00	86908.92	1054.00
	SELF-ROUTE	30.29	46.72	72.28	25389.61	962.14
	adaptive-*k*	**31.51**	69.47	71.97	27729.10	979.11

Table 13: Full Llama4-Scout's results in the HoloBench tasks.

Info amount	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
info5k	zeroshot	9.65	0.00 ± 0.00	100.00 ± 0.00	56.18	328.77
	fixed-1k	28.05	12.05 ± 6.42	99.18 ± 0.43	994.79	591.14
	fixed-5k	48.57	51.92 ± 29.82	95.85 ± 1.86	4195.84	799.80
	fixed-10k	54.30	66.68 ± 30.73	91.80 ± 2.92	8024.09	834.79
	fixed-25k	56.39	78.48 ± 26.96	79.59 ± 4.56	19625.13	874.41
	fixed-50k	53.54	86.79 ± 20.88	57.76 ± 5.92	39342.62	919.02
	full-context	55.13	100.00 ± 0.00	0.00 ± 0.00	86093.44	1101.66
	SELF-ROUTE	48.47	65.03 ± 29.24	79.91 ± 35.98	18195.21	897.80
	adaptive-*k*	**51.77**	75.74 ± 30.48	74.07 ± 25.68	24711.58	836.68
info10k	zeroshot	9.06	0.00 ± 0.00	100.00 ± 0.00	56.18	322.59
	fixed-1k	23.16	6.53 ± 3.11	99.19 ± 0.40	999.90	609.29
	fixed-5k	39.02	31.77 ± 14.86	95.84 ± 1.90	4233.97	892.93
	fixed-10k	42.91	59.10 ± 27.14	91.74 ± 3.47	8254.73	927.12
	fixed-25k	44.54	78.18 ± 26.55	79.55 ± 5.42	19898.58	883.56
	fixed-50k	50.37	87.34 ± 20.51	57.88 ± 6.41	39561.99	1020.78
	full-context	48.02	100.00 ± 0.00	0.00 ± 0.00	86347.34	1357.50
	SELF-ROUTE	37.68	44.92 ± 27.75	79.95 ± 36.00	18010.20	1053.86
	adaptive-*k*	**44.91**	68.54 ± 32.55	79.22 ± 21.59	20348.23	934.27
info25k	zeroshot	7.26	0.00 ± 0.00	100.00 ± 0.00	56.18	322.77
	fixed-1k	19.15	2.77 ± 1.15	99.21 ± 0.32	997.53	629.03
	fixed-5k	30.36	14.06 ± 5.50	95.96 ± 1.56	4226.03	896.96
	fixed-10k	30.28	28.80 ± 9.87	91.86 ± 3.09	8298.52	867.99
	fixed-25k	40.47	68.13 ± 22.77	79.60 ± 7.10	20562.19	1053.26
	fixed-50k	44.69	86.88 ± 20.14	58.46 ± 8.58	40309.13	1291.90
	full-context	43.38	100.00 ± 0.00	0.00 ± 0.00	86997.00	1455.21
	SELF-ROUTE	28.57	29.97 ± 33.04	79.00 ± 36.97	19123.37	1038.17
	adaptive-*k*	**39.40**	66.16 ± 36.90	73.86 ± 23.04	25958.96	1122.81
info50k	zeroshot	8.51	0.00 ± 0.00	100.00 ± 0.00	56.18	342.58
	fixed-1k	11.24	1.47 ± 0.55	99.22 ± 0.23	1004.93	574.83
	fixed-5k	22.39	7.54 ± 2.52	96.02 ± 1.12	4246.40	783.99
	fixed-10k	24.53	15.39 ± 4.25	92.01 ± 2.13	8317.29	825.16
	fixed-25k	27.94	39.55 ± 8.89	79.75 ± 5.38	20706.06	1127.69
	fixed-50k	34.89	76.90 ± 16.82	58.85 ± 9.99	41427.09	1317.22
	full-context	36.54	100.00 ± 0.00	0.00 ± 0.00	88197.90	1706.96
	SELF-ROUTE	21.90	24.17 ± 35.52	78.99 ± 36.95	19460.93	1183.29
	adaptive-*k*	**34.63**	67.43 ± 38.13	60.73 ± 25.94	39897.64	1687.78
Average	zeroshot	8.62	0.00	0.00	56.18	329.18
	fixed-1k	20.40	5.70	99.20	999.29	601.07
	fixed-5k	35.09	26.32	95.92	4225.56	843.42
	fixed-10k	38.00	42.49	91.85	8223.66	863.76
	fixed-25k	42.33	66.09	79.62	20197.99	984.73
	fixed-50k	45.87	84.48	58.24	40160.21	1137.23
	full-context	45.77	100.00	0.00	86908.92	1405.33
	SELF-ROUTE	34.16	41.02	79.46	18697.43	1043.28
	adaptive-*k*	**42.68**	69.47	71.97	27729.10	1145.38

Table 14: Full Llama4-Maverick's results in the HoloBench tasks.

A.3.3 Factoid QA tasks (GTE embeddings)

Task	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
HotpotQA	zeroshot	50	0.00 ± 0.00	100.00 ± 0.00	63.03 ± 6.98	19.31 ± 16.53
	fixed-1k	50	6.00 ± 16.33	99.16 ± 0.20	881.29 ± 76.42	31.19 ± 16.09
	fixed-5k	51	9.33 ± 19.44	95.84 ± 0.67	4106.52 ± 327.20	30.61 ± 16.31
	fixed-10k	54	13.67 ± 23.73	91.81 ± 1.19	8145.75 ± 576.29	29.29 ± 15.20
	fixed-25k	58	27.50 ± 32.17	80.21 ± 2.40	20337.90 ± 1369.28	29.06 ± 17.08
	fixed-50k	60	43.50 ± 33.16	61.79 ± 3.81	40520.53 ± 2661.25	28.96 ± 18.42
	full-context	48	100.00 ± 0.00	0.00 ± 0.00	109666.92 ± 5537.59	18.91 ± 20.30
	SELF-ROUTE	46	25.83 ± 39.17	78.64 ± 37.04	76201.89 ± 52199.49	22.69 ± 20.51
	adaptive-*k*	**49**	5.50 ± 15.72	99.20 ± 0.36	877.06 ± 409.16	32.87 ± 23.82
NQ	zeroshot	57	0.00 ± 0.00	100.00 ± 0.00	53.37 ± 2.29	27.38 ± 24.23
	fixed-1k	53	2.65 ± 9.56	99.29 ± 0.19	813.57 ± 84.46	31.61 ± 25.74
	fixed-5k	53	9.07 ± 18.72	96.44 ± 0.49	3871.35 ± 353.77	33.67 ± 28.72
	fixed-10k	58	14.22 ± 22.67	92.96 ± 0.77	7702.16 ± 668.50	34.24 ± 26.91
	fixed-25k	62	31.38 ± 31.59	82.54 ± 1.61	19183.81 ± 1661.81	35.55 ± 29.97
	fixed-50k	64	45.18 ± 33.28	65.32 ± 2.88	38288.26 ± 3141.73	36.84 ± 34.84
	full-context	41	100.00 ± 0.00	0.00 ± 0.00	110607.54 ± 4711.16	24.74 ± 33.79
	SELF-ROUTE	49	20.67 ± 31.77	78.12 ± 38.03	54951.43 ± 55725.18	27.13 ± 25.83
	adaptive-*k*	**51**	2.85 ± 9.72	99.22 ± 0.24	907.79 ± 263.30	32.75 ± 25.98
TriviaQA	zeroshot	91	0.00 ± 0.00	100.00 ± 0.00	60.09 ± 7.79	8.33 ± 7.16
	fixed-1k	96	3.00 ± 13.48	99.15 ± 0.49	857.89 ± 96.49	16.98 ± 10.66
	fixed-5k	94	4.79 ± 15.73	95.94 ± 0.83	4040.80 ± 358.14	17.35 ± 10.37
	fixed-10k	93	6.43 ± 17.79	92.04 ± 1.25	8038.26 ± 670.72	17.64 ± 10.47
	fixed-25k	96	16.65 ± 30.03	80.93 ± 1.99	19985.92 ± 1507.53	16.64 ± 10.50
	fixed-50k	95	39.57 ± 41.22	63.01 ± 3.13	39806.06 ± 2867.46	17.17 ± 10.11
	full-context	62	100.00 ± 0.00	0.00 ± 0.00	110733.69 ± 3419.97	12.94 ± 12.48
	SELF-ROUTE	88	10.20 ± 24.97	87.30 ± 27.60	32620.05 ± 48431.50	13.76 ± 10.53
	adaptive-*k*	**93**	3.00 ± 13.48	99.16 ± 0.56	929.99 ± 604.34	17.33 ± 10.55
Average	zeroshot	66.00	0.00	0.00	58.83	18.34
	fixed-1k	66.33	3.88	99.20	850.92	26.59
	fixed-5k	66.00	7.73	96.07	4006.22	27.21
	fixed-10k	68.33	11.44	92.27	7962.06	27.06
	fixed-25k	72.00	25.18	81.23	19835.88	27.08
	fixed-50k	73.00	42.75	63.38	39538.28	27.66
	full-context	50.33	100.00	0.00	110336.05	18.86
	SELF-ROUTE	61.00	18.90	81.35	54591.12	21.19
	adaptive-*k*	**64.33**	3.78	99.19	904.95	27.65

Table 15: Full GPT-4o's results in the factoid QA tasks with the embeddings by gte-Qwen2-1.5B-instruct.

A.3.4 HoloBench (GTE embeddings)

Info amount	Method	Score	Context recall	Reduction (%)	n_{in}	n_{out}
	zeroshot	7.22	0.00 ± 0.00	100.00 ± 0.00	58.13	65.21
	fixed-1k	26.14	12.05 ± 6.42	99.18 ± 0.43	1000.07	289.39
	fixed-5k	42.32	51.92 ± 29.82	95.85 ± 1.86	4194.01	913.32
	fixed-10k	49.79	66.68 ± 30.73	91.80 ± 2.92	8011.64	1334.67
info5k	fixed-25k	46.27	78.48 ± 26.96	79.59 ± 4.56	19574.94	2087.61
	fixed-50k	43.82	86.79 ± 20.88	57.76 ± 5.92	39224.80	3188.69
	full-context	48.30	100.00 ± 0.00	0.00 ± 0.00	73652.50	3680.13
	SELF-ROUTE	40.69	65.18 ± 28.08	76.14 ± 38.33	12113.12	1288.40
	adaptive-*k*	**45.23**	82.20 ± 25.15	64.79 ± 32.49	29079.80	1879.49
	zeroshot	5.11	0.00 ± 0.00	100.00 ± 0.00	58.13	64.00
	fixed-1k	21.83	6.53 ± 3.11	99.19 ± 0.40	1003.83	298.53
	fixed-5k	32.45	31.77 ± 14.86	95.84 ± 1.90	4228.71	1001.83
	fixed-10k	36.48	59.10 ± 27.14	91.74 ± 3.47	8235.02	2589.58
info10k	fixed-25k	39.65	78.18 ± 26.55	79.55 ± 5.42	19838.92	3527.68
	fixed-50k	38.55	87.34 ± 20.51	57.88 ± 6.41	39437.42	4061.82
	full-context	41.75	100.00 ± 0.00	0.00 ± 0.00	75768.00	4767.44
	SELF-ROUTE	32.28	45.87 ± 24.77	79.47 ± 35.80	18646.59	1307.47
	adaptive-*k*	**39.66**	78.99 ± 28.37	65.70 ± 30.96	28475.07	2238.23
	zeroshot	3.54	0.00 ± 0.00	100.00 ± 0.00	58.13	65.69
	fixed-1k	16.51	2.77 ± 1.15	99.21 ± 0.32	999.62	282.18
	fixed-5k	27.52	14.06 ± 5.50	95.96 ± 1.56	4215.72	1350.80
	fixed-10k	29.10	28.80 ± 9.87	91.86 ± 3.09	8269.06	2520.64
info25k	fixed-25k	40.25	68.13 ± 22.77	79.60 ± 7.10	20475.06	3406.40
	fixed-50k	34.18	86.88 ± 20.14	58.46 ± 8.58	40152.73	4366.80
	full-context	42.24	100.00 ± 0.00	0.00 ± 0.00	75787.37	4802.50
	SELF-ROUTE	23.60	31.02 ± 32.35	77.63 ± 37.70	18905.02	1334.77
	adaptive-*k*	**36.33**	76.47 ± 31.62	58.43 ± 29.94	36711.21	3509.82
	zeroshot	5.19	0.00 ± 0.00	100.00 ± 0.00	58.13	62.30
	fixed-1k	11.59	1.47 ± 0.55	99.22 ± 0.23	1006.91	274.89
	fixed-5k	20.62	7.54 ± 2.52	96.02 ± 1.12	4235.36	735.80
	fixed-10k	23.15	15.39 ± 4.25	92.01 ± 2.13	8288.68	1595.42
info50k	fixed-25k	28.11	39.55 ± 8.89	79.75 ± 5.38	20622.82	4269.89
	fixed-50k	34.58	76.90 ± 16.82	58.85 ± 9.99	41264.63	4244.67
	full-context	27.40	100.00 ± 0.00	0.00 ± 0.00	87936.41	3051.19
	SELF-ROUTE	22.08	21.90 ± 31.81	79.88 ± 35.94	17563.86	1843.27
	adaptive-*k*	**30.80**	72.54 ± 36.80	49.06 ± 29.83	46590.91	3806.21
	zeroshot	5.26	0.00	0.00	58.13	64.30
	fixed-1k	19.02	5.70	99.20	1002.61	286.25
	fixed-5k	30.73	26.32	95.92	4218.45	1000.44
	fixed-10k	34.63	42.49	91.85	8201.10	2010.08
Average	fixed-25k	38.57	66.09	79.62	20127.94	3322.89
	fixed-50k	37.78	84.48	58.24	40019.90	3965.49
	full-context	39.92	100.00	0.00	78286.07	4075.32
	SELF-ROUTE	29.66	40.99	78.28	16807.15	1443.48
	adaptive-*k*	**38.00**	77.55	59.49	35214.25	2858.44

Table 16: Full GPT-4o's results in the HoloBench tasks with the embeddings by gte-Qwen2-1.5B-Instruct.