MemSearch-o1: Empowering Large Language Models with Reasoning-Aligned Memory Growth in Agentic Search

Anonymous ACL submission

Abstract

Recent advances in large language models (LLMs) have scaled the potential for reasoning and agentic search, wherein models autonomously plan, retrieve, and reason over external knowledge to answer complex queries. However, the iterative think–search loop accumulates long system memories, leading to memory dilution problem. In addition, existing memory management methods struggle to capture fine-grained semantic relations between queries and documents and often lose substantial information. Therefore, we propose **MemSearch-o1**, an agentic search framework built on reasoning-aligned memory growth and retracing. MemSearch-o1 dynamically grows fine-grained memory fragments from memory seed tokens from the queries, then retraces and deeply refines the memory via a contribution function, and finally reorganizes a globally connected memory path. This shifts memory management from stream-like concatenation to structured, token-level growth with path-based reasoning. Experiments on eight benchmark datasets show that MemSearch-o1 substantially mitigates memory dilution, and more effectively activates the reasoning potential of diverse LLMs, establishing a solid foundation for memory-aware agentic intelligence. Our code is available at `https://anonymous.4open.science/r/MemSearch-o1-2BF3`.

1 Introduction

In recent years, retrieval-augmented generation (RAG) (Lewis et al., 2020) has emerged as a powerful framework that enables large language models (LLMs) to access external corpora by retrieving text chunks relevant to a given query (Zhang et al., 2025b). While effective in enhancing LLMs with factual knowledge, RAG often provides shallow support: retrieved passages are limited in scope, and the pipeline lacks explicit reasoning over the original query. This significantly constrains performance on complex, multi-hop problems. To overcome these limitations, the paradigm of deep search has been proposed (Wu et al., 2025; Li et al., 2025). Unlike conventional RAG, deep search autonomously plans, retrieves, reflects, and reasons over external knowledge in an iterative manner, constructing deeper reasoning chains through repeated interactions with knowledge sources (Jin et al., 2025; Li et al., 2025). Finally, deep search is especially well-suited to LLMs with advanced reasoning capabilities (Wang, 2025; Liu et al., 2024; Yang et al., 2025), which effectively integrates and exploits the retrieved knowledge.

Despite its promise, the deep search paradigm still faces two critical limitations. First, the accumulated thinking history and redundant document fragments often introduce irrelevant information (Yan et al., 2025; Luo et al., 2025; Yu et al., 2025), while attention dilution in LLMs causes much of the context to be overlooked (Liu et al., 2023). As a result, key evidence may remain undiscovered, ultimately degrading reasoning quality. This issue becomes more severe as system memory grows: the signal-to-noise ratio declines, making it increasingly difficult for the model to focus on query goals. Second, although advanced LLMs possess substantial latent reasoning capacity enabled by their scale (Kaplan et al., 2020), current deep search studies have yet to fully exploit this potential (Wu et al., 2025). In addition, since memory dilution cannot be solved by prompt engineering alone, effective explicit memory management strategies are essential to unlock the reasoning capabilities of LLMs.

Prior efforts to address memory dilution in deep search can be broadly divided into two paradigms. The first focuses on memory summarization and refinement, where retrieved documents are compressed (Li et al., 2025; Chhikara et al., 2025) or filtered (Qin et al., 2025) to improve reasoning quality. While intuitive, these methods often fail to adapt to the evolving semantics of updated queries, resulting in the loss of critical information. The

second paradigm centers on memory pruning and distillation, where the system selectively retains or compresses past context for subsequent reasoning through designing special prompts (Zhou et al., 2025; Yan et al., 2025). However, these approaches largely optimize for overall task rewards without explicitly modeling the structure and logic of stored memories, leading to suboptimal reasoning paths. Even methods that explicitly retrieve key elements from refined memories (Xu et al., 2025) risk discarding important connections, as their quadratic search mechanisms may overlook latent semantic dependencies across memory fragments.

To overcome these challenges, we propose **MemSearch-o1**, a novel agentic search framework that emphasizes fine-grained, reasoning-aligned memory growth. Instead of refining memories based on complex query-document semantic relations, our approach expands memories by collecting contents associated with specific tokens of subject, action, degree, temporal (Szabó, 2015), i.e., seed tokens of the query, thereby aligning more closely with the search goal. The framework then retraces and filters these token-associated memories to build a semantically coherent and logically structured memory path for answer generation. By enabling the agent to operate within compact, query-focused memory spaces, rather than diluted long contexts, MemSearch-o1 provides a concise yet comprehensive semantic environment optimized for reasoning. This targeted memory management substantially enhances the depth, precision, and overall quality of the deep search.

Our main contributions are as follows:

- We propose MemSearch-o1, the first deep search framework that grows fine-grained memory fragments from query tokens, enabling clearer semantic alignment and stronger reasoning.
- We develop a retracing-based memory refinement mechanism that filters and reorganizes fragments into concise, coherent memory paths optimized for multi-hop reasoning.
- Extensive experiments on eight QA benchmarks and LLMs show that MemSearch-o1 effectively mitigates memory dilution and consistently outperforms strong baselines.

2 Method

In this section, we will introduce the MemSearch-o1 in detail. We first give an overview of our framework (Section 2.1), then introduce the memory seeds extraction from queries (Section 2.2), followed by memory fragments growth and memory path retracing (Section 2.3 and 2.4).

2.1 Overview

Most agentic search frameworks produce especially long system memories during the reasoning process, leading to a serious memory dilution problem (Wu et al., 2025; Liu et al., 2023). Although existing memory management methods can refine the redundant memories (Shi et al., 2025; Chhikara et al., 2025), they only focus on the semantic relevance to the query, which may either be too complex for LLMs to comprehend or lead to a dramatic information loss. To tackle this issue, we develop MemSearch-o1, an agentic search framework that employs reasoning-aligned memory growth and retrace for LLMs with reasoning ability.

The overall framework is shown in Figure 1. The MemSearch-o1 starts from an original search query q_o, and a predefined instruction for LLM reasoning. Then, the LLM raises a new required query for search, and we utilize this query for **memory seeds preparation**. Specifically, the raised query will be split into memory seeds, and each seed consists of several memory seed tokens grouped by their parts of speech (Brown, 1957; Szabó, 2015). After that is the **memory fragments growth** stage, where relevant documents are retrieved using the latest raised query, and the memory seed tokens are fed into the LLM to guide the information extraction. The grown fragment sentences are then conveyed into the LLM again for next-round reasoning. The LLM reasoning round, together with corresponding memory growth, is iteratable. Finally, we collect all the fragment sentences for **memory path retracing** to deeply refine the memory. We design a contribution function to select fragments with high relevance to the original query and high bridge potential with other fragments. Finally, we use greedy search to find a semantically smooth memory path for the final generation. The whole framework improves the capability of our deep search system to exploit essential information for concise yet semantically rich memory growth and perform higher-quality reasoning and generation.

2.2 Memory Seeds Preparation

Most existing memory management approaches rely on LLMs to refine retrieved documents or to directly fetch relevant fragments from stored mem-

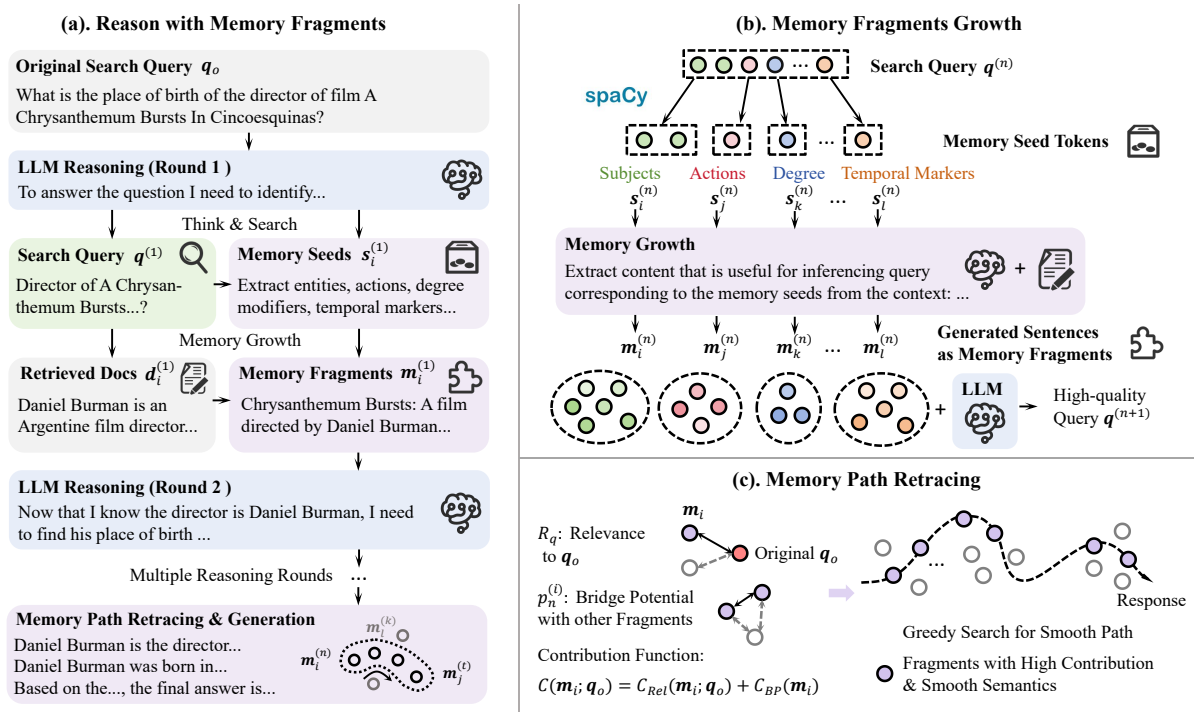


Figure 1: Overview Framework of MemSearch-o1.

ories, but such strategies often incur significant information loss (Wang et al., 2025b). To tackle this issue, our method grows fine-grained memories from seed tokens extracted from queries, enabling the construction of a concise yet semantically rich memory space that remains closely aligned with the evolving query goals.

Inspired by linguistics, a sentence is composed of words with eight parts of speech, among which *nouns, pronouns, verbs, adjectives*, and *adverbs* usually contain the richest semantics (Szabó, 2015). In the context of deep search and memory growth, nouns and pronouns often denote **subjects** such as people, places, and objects, as well as **temporal markers** that indicate the time or duration of events. Verbs, by contrast, usually capture the **actions** performed by or involving these subjects, providing essential links within reasoning chains. Adjectives and adverbs function as **degree modifiers**, describing the properties of subjects or the manner and intensity of actions. Based on these observations, we categorize nouns and pronouns into subjects and temporal markers, and group adjectives and adverbs together as degree modifiers. This decomposition serves as the foundation for our token-level memory growth strategy, enabling fine-grained alignment between the goals of queries and memory exploration.

Specifically, let $\boldsymbol{q}^{(n)} = [q_1^{(n)}, q_2^{(n)}, \ldots, q_L^{(n)}]$ denote a search query consisting of L tokens at the n-th turn. We then identify and partition $\boldsymbol{q}^{(n)}$ into L_r memory seeds as follows:

$$\mathcal{S}^{(n)} = \{\boldsymbol{s}_1^{(n)}, \boldsymbol{s}_2^{(n)}, \ldots, \boldsymbol{s}_{L_r}^{(n)}\} \leftarrow \boldsymbol{q}^{(n)}, \quad (1)$$

where each seed $\boldsymbol{s}_i^{(n)} = \{q_j^{(n)}\}_{j=1}^{z_i}$ contains z_i tokens from $\boldsymbol{q}^{(n)}$ and corresponds to one type from subjects, actions, degree modifiers, and temporal markers. For efficient implementation, we use the spaCy[1] toolkit to identify and group these tokens into respective memory seeds.

2.3 Memory Fragments Growth

Although existing system memory refinement approaches can extract key information from retrieved documents, the complex relationships between queries and documents can degrade the quality of both subsequent queries and final answers (Lee, 2025). In particular, the smooth semantic nature of sentence embeddings in vector space causes query representations and lengthy document embeddings to become highly entangled. This entanglement introduces irrelevant information and obscures critical content, thereby reducing the effectiveness of memory refinement. Furthermore, in agentic search, such degraded memories are repeatedly used as inputs for subsequent reasoning

[1]https://spacy.io/

steps, compounding errors and leading to a gradual decline in downstream reasoning quality.

To address this issue, we grow memory fragments that contain the contents of memory seeds introduced in Section 2.2 based on the retrieved information. Instead of comprehending complex semantic relations between the query and retrieved texts, memory seeds guide LLMs to extract diverse and useful summaries aligned with the query goal. Concretely, we provide the LLM with the memory seeds $\boldsymbol{s}_i^{(n)}$, retrieval results $\boldsymbol{D}^{(n)}$ of the n-th reasoning round and task instructions $\boldsymbol{I}_M$, and prompt it to expand each seed into diverse yet relevant memory fragments for subsequent reasoning. Thus, the generation probability P_M of the memory fragment $\boldsymbol{M}^{(n)}$ can be formulated as:

$$P_M = \prod_{i=1}^{L_r} \prod_{t=T_{s_{i-1}}^{(n)}}^{T_{s_i}^{(n)}} P(\boldsymbol{M}_t^{(n)} | \boldsymbol{M}_{<t}^{(n)}, \boldsymbol{s}_i^{(n)}, \boldsymbol{I}_M, \boldsymbol{D}^{(n)}), \tag{2}$$

where $T_{s_i}^{(n)}$ is the position where the i-th memory fragment ends, and $T_{s_0}^{(n)} = 0$. $\boldsymbol{M}_t^n$ denotes the t-th generated token of the memory fragments, and $\boldsymbol{M}_{<t}^{(n)}$ is the sequence of the LLM generated before position t.

Obviously, the $\boldsymbol{s}_i^{(n)}$ serves as a seed that attracts the attention of LLMs to focus on the relevant information to it, which brings about more concise and accurate fragment contents. Subsequently, the generated tokens in Equation 2 are concatenated to form memory fragments, as formalized below.

$$\boldsymbol{m}_i^{(n)} = \boldsymbol{M}_{T_{s_{i-1}}^{(n)}:T_{s_i}^{(n)}}^{(n)}, \tag{3}$$

where $\boldsymbol{m}_i^{(n)}$ is the i-th grown memory fragment during the n-th reasoning round, and $T_{s_0}^{(n)} = 0$.

2.4 Memory Path Retracing

Existing memory management often preserves verbose traces accumulated across multiple reasoning rounds without effective integration, and answering over elongated paths exacerbates memory dilution (Shi et al., 2024). We address this by retracing the memory history and reorganizing fragments into a coherent path: MemSearch-o1 selects the most relevant evidence and assembles a semantically smooth memory trajectory, reducing redundancy and sharpening focus for reliable reasoning.

After the LLM completes search and reasoning, we retrace and collect all the memory fragments $\boldsymbol{m}_i$ from every reasoning round within a memory region. However, because search queries may drift semantically over reasoning rounds, the grown fragments can contain irrelevant information. To refine them, we design a contribution function $C(\boldsymbol{m}_i; \boldsymbol{q}_o)$ that measures both their relevance to the original query $\boldsymbol{q}_o$ and their bridge potential to other fragments. Specifically, we define the relevance contribution C_{Rel} to filter out memories that lie far from the ideal reasoning region as follows:

$$C_{Rel} = \text{Sim}(\text{Emb}(\boldsymbol{m}_i), \text{Emb}(\boldsymbol{q}_o)), \tag{4}$$

where $\text{Emb}(\cdot)$ is the sentence transformer that maps input tokens into embedding vectors, and $\text{Sim}(\cdot)$ is the cosine similarity. Memory fragments with high contribution C_{Rel} have more relevance to $\boldsymbol{q}_o$. In addition, we consider the potential of each memory fragment to connect with others through a bridge potential function:

$$C_{BP} = \frac{\sum_{j \neq i} \text{sim}(\boldsymbol{m}_i, \boldsymbol{m}_j) \cdot \sigma(U_{Rel} - \tau_s)}{\sum_{j \neq i} \sigma(U_{Rel} - \tau_s)}, \tag{5}$$

where $U_{Rel} = \max\{\text{Sim}(\boldsymbol{m}_j, \boldsymbol{q}^{(n)})\}_{j=1}^N$ is the upper bound of the relevance between memory fragments and search query, and N is the total number of reasoning rounds. Threshold τ_s together with the ReLU function $\sigma(\cdot)$ restricts the contribution score of fragments with the lowest relevance to the search query in reasoning. The maximum relevance $\sigma(U_{Rel} - \tau_s)$ serves as the weight on the bridge potential with other fragments, which is calculated by the cosine similarity function. Then we can derive the contribution score of the memory fragment $\boldsymbol{m_i}$ as:

$$C(\boldsymbol{m}_i; \boldsymbol{q}_o) = \alpha \cdot C_{Rel} + \beta \cdot C_{BP}, \tag{6}$$

where α and β are predefined weights. Then we need to reorganize the fragments that we retrace and select in the memory region:

$$\mathcal{M}_q = \left\{ \boldsymbol{m}_i \in \mathbb{R}^d | \, C(\boldsymbol{m}_i; \boldsymbol{q}_o) > \tau_r \right\}, \tag{7}$$

where τ_r is the threshold that filters the memory fragments with low contribution, i.e., with low semantic relevance to queries and other memory fragments. In order to find a semantically smooth memory path tailored for reasoning, we aim to solve the following optimization problem:

$$\mathcal{P}^* = \arg\max_{\mathcal{P}} \sum_{k=1}^{|\mathcal{M}_q|} C(\boldsymbol{m}_{i_k}; \boldsymbol{q}_o) \cdot \mu(\boldsymbol{m}_{i_k}),$$
$$\mu(\boldsymbol{m}_{i_k}) = \exp\left(-\lambda(1 - \text{Sim}(\boldsymbol{m}_{i_k}, \boldsymbol{m}_{i_{k-1}}))\right), \tag{8}$$

where $\mathcal{P} = (\boldsymbol{m}_{i_1}, \boldsymbol{m}_{i_2}, ..., \boldsymbol{m}_{i_K})$ is the ideal memory path consisting of K maximum fragments, and $\mu(\boldsymbol{m}_{i_k})$ is the penalty function that ensures the smooth semantics between the current fragment and connected fragment. In this work, we utilize the greedy search strategy (Chickering, 2002) to find the memory path. Finally, the memory path is directly used for answer generation.

Since the memory fragments are generated based on prior reasoning steps, and we have already organized them through a path that incorporates recall and semantic smoothing, during the generation phase, we no longer need to rely on the original system memory. Instead, we can generate answers directly from the constructed memory path. The detailed deep search algorithm of MemSearch-o1 inference is displayed in Appendix A.1.

3 Experiment

3.1 Dataset and Metrics

To evaluate the effectiveness of MemSearch-o1, we conduct extensive experiments on eight benchmark datasets from LongBench (Bai et al., 2023), covering both multi-document and single-document QA. The multi-document tasks include HotpotQA (Yang et al., 2018), 2WikiMQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022), DuReader (He et al., 2017), while the single-document tasks include NarrativeQA (Kočiskỳ et al., 2018), Qasper (Dasigi et al., 2021), and Multi-FieldQA (Bai et al., 2023). In these settings, the long contexts of QA tasks serve as the corpus for external knowledge retrieval, and the retrieved top-k passages are substantially shorter than the average document length in the corpus. We focus on both multi-document and single-document QA to comprehensively evaluate the performance of competing methods. Detailed dataset statistics are provided in Appendix A.2. Following prior work (Bai et al., 2023), we report QA-F1 as the primary evaluation metric for all datasets except DuReader, which is evaluated using ROUGE-L.

3.2 Baselines

To verify the effectiveness of our MemSearch-o1 framework, we compare the performance with the following baselines: (1) **Standard RAG** (Lewis et al., 2020), (2) Deep search without memory management: **Search-o1 (RAgent)** (Li et al., 2025) (3) Deep search with memory management: **Search-o1 (Refined)** (Li et al., 2025), **MemoryBank** (Zhong et al., 2024), **A-Mem** (Xu et al., 2025), and **Amber** (Qin et al., 2025). The detailed information of the baselines and the backbone models can be found in Appendix A.3.

3.3 Implementation Details

For all retrieval processes in the baselines, including standard RAG and deep search systems, we use BGE-M3 (Chen et al., 2024) as both the retriever and sentence embedding model. The maximum number of search rounds is fixed at $N = 5$, with the top-k documents ($k = 3$) retrieved in chunks of 256 tokens. We set filtering thresholds τ_s and τ_r to 0.3 to discard memory fragments weakly related to either the original query or reasoning-driven search queries. In the contribution function, weights are set to $\alpha = 0.6$ and $\beta = 0.4$, while memory reorganization applies a penalty $\lambda = 1$ to ensure semantic smoothness and restricts each path to $K = 10$ fragments. Following prior work (Li et al., 2025; Wang et al., 2025a), we evaluate MemSearch-o1 alongside baselines using three backbone LLMs: Qwen2.5-72B-Instruct (Yang et al., 2024), DeepSeek V3.1 (Liu et al., 2024), and GPT-4o (Hurst et al., 2024). Full prompt templates are provided in Appendix A.6.

3.4 Overall Performance

As shown in Table 1, MemSearch-o1 achieves state-of-the-art results across both multi- and single-document QA benchmarks, outperforming RAG and advanced deep search methods such as Amber and A-Mem under all three LLM backbones. The gains are especially notable on complex reasoning tasks: with DeepSeek V3.1, MemSearch-o1 improves over the strongest baseline by 21.93% on HotpotQA, 16.53% on 2WikiMQA, and 17.38% on MuSiQue. These results demonstrate that fine-grained memory growth from seed tokens and retraced memory paths substantially enhance reasoning quality. To further show that MemSearch-o1 relieves the memory dilution problem and constructs concise memory paths for generation, we measure the number of tokens in the system memory. Detailed analysis can be found in Appendix A.4.

Deep search with memory management generally surpasses typical RAG on multi-document QA, as iterative retrieval supports multi-hop reasoning. Yet, existing memory strategies remain limited. Summarization-based methods such as Search-o1, MemoryBank, and A-Mem often lose key evidence, while Amber mitigates memory dilution through

Models	Methods	MultidocQA				Singledoc QA				Avg.
		HotpotQA	2WikiMQA	MuSiQue	DuReader	NarrativeQA	Qasper	MultiField-en	MultiField-zh	
Qwen 2.5-72B-Instruct	Direct RAG	54.40	47.23	27.54	27.49	**17.73**	35.94	45.43	52.29	38.51
	Search-o1 (RAgent)	57.41	62.86	**44.78**	18.91	14.37	22.88	37.77	49.97	38.62
	Search-o1 (Refined)	52.11	35.74	37.99	15.30	16.12	17.51	33.79	44.45	31.63
	MemoryBank	47.27	44.89	28.26	19.24	16.26	26.28	32.42	43.46	32.26
	A-Mem	54.24	60.08	39.95	18.74	15.18	22.09	34.59	50.79	36.96
	Amber	53.78	61.16	37.61	20.57	16.65	27.19	37.98	52.26	38.40
	MemSearch-o1	**59.71***	**65.95***	44.11	**32.06***	17.30	**36.18***	**49.05***	**59.91***	**45.53***
Deepseek V3.1	Direct RAG	53.98	43.76	26.95	18.72	21.48	32.45	43.41	48.15	36.11
	Search-o1 (RAgent)	54.64	56.14	44.31	15.28	20.59	33.04	38.67	49.76	39.05
	Search-o1 (Refined)	48.46	36.75	33.32	16.12	18.59	22.97	33.82	44.79	31.85
	MemoryBank	47.51	46.65	38.05	17.57	21.83	28.60	33.34	42.33	34.49
	A-Mem	49.57	41.38	25.67	17.10	20.92	27.55	28.27	47.29	32.22
	Amber	55.59	58.63	41.29	18.39	22.48	32.86	39.27	49.31	39.79
	MemSearch-o1	**67.78***	**68.32***	**52.01***	**27.23***	**23.04***	**37.94***	**52.26***	**56.81***	**48.17***
GPT-4o	Direct RAG	47.93	42.73	24.19	15.43	16.84	34.56	44.34	49.71	34.47
	Search-o1 (RAgent)	54.67	56.15	37.57	18.01	15.46	28.90	39.85	48.54	37.39
	Search-o1 (Refined)	52.45	35.64	18.23	14.49	11.69	19.04	38.98	36.77	28.41
	MemoryBank	52.51	44.07	29.50	19.85	18.76	32.83	37.74	41.61	34.61
	A-Mem	53.04	50.42	34.17	18.22	18.43	24.95	39.59	45.32	35.52
	Amber	55.05	57.44	37.38	21.27	18.23	31.42	40.31	49.79	38.86
	MemSearch-o1	**56.61***	**59.58***	**39.88***	**27.12***	**19.19***	**36.87***	**48.36***	**53.24***	**42.61***

Table 1: Overall Performance Comparison on MultidocQA and Singledoc QA Benchmarks. The best results are bolded and the second best results are underlined. "*" indicates the statistically significant improvements (i.e.,two-sided t-test with $p < 0.05$) over the baselines.

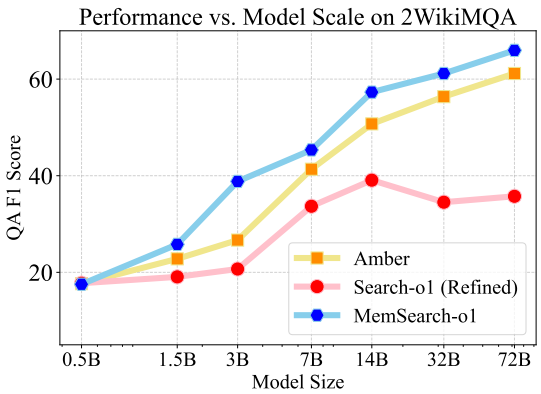


Figure 2: Model Size Scaling on MemSearch-o1

chunk- and sentence-level refinement but still struggles to capture deeper logical connections.

On tasks with localized evidence (e.g., Qasper, MultiFieldQA-en), standard RAG can be competitive, since these scenarios require less iterative retrieval. Many deep search methods over-search here, leading to instability. In contrast, MemSearch-o1 avoids this issue by expanding memory from seed tokens, ensuring sufficient coverage while maintaining concise and precise reasoning paths. Together, these results establish MemSearch-o1 as a robust framework for deep search reasoning.

3.5 Scaling Law of MemSearch-o1

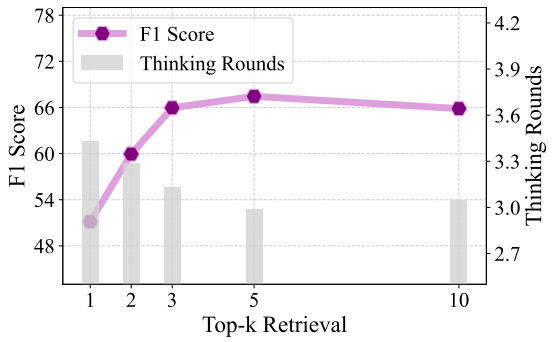
In this experiment, we employ the Qwen2.5 instruct model series from 0.5B to 72B to perform the model size scaling and evaluate the performance on the 2WikiMQA dataset. The corresponding results of MemSearch-o1 are shown in Figure 2.

Smaller models struggle to accurately follow search instructions and thus fail to perform deep reasoning without additional training. As model size increases, however, reasoning capabilities are activated more rapidly. Around the 3B scale, our method effectively unlocks deep search-based reasoning, and larger models achieve even higher accuracy, confirming the strong compatibility of our approach with reasoning tasks. In contrast, Amber shows delayed activation, with noticeable improvements only beyond the 7B scale, and still lags behind MemSearch-o1 in overall performance. Search-o1 (Refined), meanwhile, exhibits unstable behavior: after an initial boost, its performance fluctuates on larger models and fails to scale consistently. This instability reflects its reliance on complex semantic associations between queries and retrieved documents, which often leads to information loss and constrains its reasoning capacity.

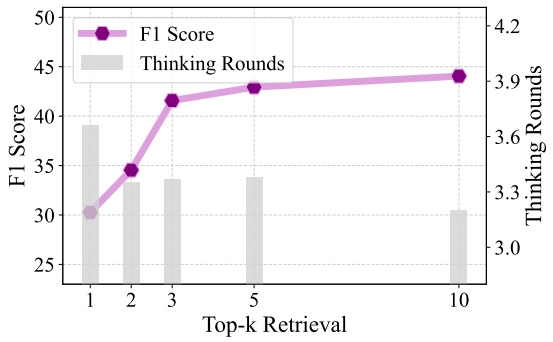
3.6 Top-k Scalability for Retrieval

In this subsection, we vary the number of top-k documents retrieved during the agentic search process to examine how retrieval volume influences both the performance of MemSearch-o1 and the average number of reasoning steps. Experiments are conducted with Qwen2.5-72B-Instruct on the 2WikiMQA and MuSiQue datasets, and results are shown in Figure 3.

Two distinct patterns emerge. As illustrated in



(a) 2WikiMQA



(b) MuSiQue

Figure 3: Top-k Scalability for MemSearch-o1.

Figure 3(a), increasing k initially enriches the memory with more relevant information, reducing the number of reasoning steps required. However, excessively large k introduces redundancy, diluting useful context and ultimately reducing accuracy while increasing the number of search rounds. By contrast, in Figure 3(b), performance continues to improve as k grows. For datasets like MuSiQue, where relevant evidence is dispersed across many chunks, larger retrieval sets provide greater information gains for memory construction. Therefore, even though lengthy documents contain abundant information, MemSearch-o1, by employing memory-seed-guided memory growth and focusing on information extraction at the token level, can still distill more effective content from extended documents, thereby reducing the required number of reasoning steps accordingly. Detailed values and additional analysis are provided in Appendix A.5.

3.7 Ablation Study

We conduct the ablation study and remove the memory management strategy step by step. The results are shown in Table 2. Specifically, we remove the memory path retracing and reorganization (*w/o memory retracing*) first, to verify the effectiveness of deep refinement in Memsearch-o1. Then, based on this, we remove the memory seeds and fragments growth (*w/o memory*), to show the efficacy of memory management.

Dataset	w/o memory	w/o retracing	Complete
hotpotqa	54.64	64.27	**67.78**
2wikimqa	56.41	63.91	**68.32**
musique	44.31	47.69	**52.01**
dureader	15.28	25.05	**27.23**
narrativeqa	20.59	22.50	**22.84**
qasper	33.04	35.30	**37.94**
multifieldqa-en	38.67	46.33	**52.26**
multifieldqa-zh	49.76	53.73	**56.81**

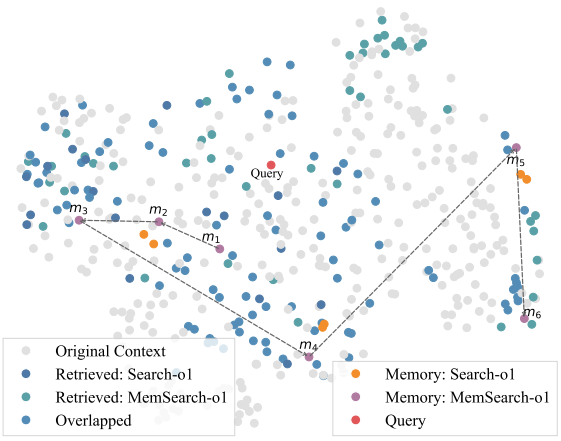
Table 2: Ablation Study of MemSearch-o1

The ablation results highlight the importance of memory management in MemSearch-o1. When memory is completely removed, the LLM is exposed to redundant document content, and the complex semantic relations between queries and retrieved texts lead to confused reasoning, causing a noticeable performance drop. Removing the memory path retracing also degrades performance, as unorganized memory fragments accumulate irrelevant information with increasing semantic distance from the original query. These findings confirm that both memory growth and retracing are essential for maintaining reasoning accuracy.

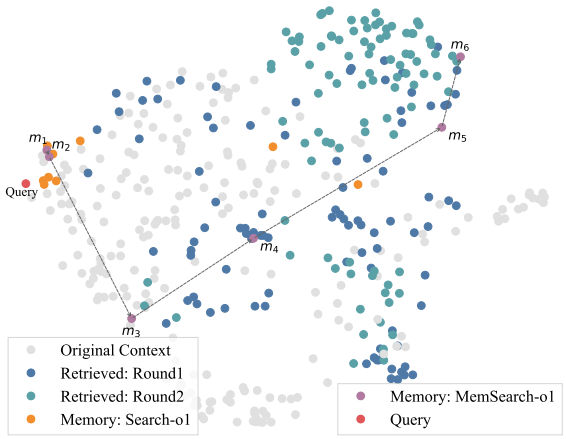
3.8 Memory Path Vizualization

In this subsection, we present case studies to illustrate how MemSearch-o1 reasons along memory paths, providing an intuitive demonstration of its effectiveness. We compare our method with Search-o1 (Refined) using the Qwen2.5-72B-Instruct backbone. Memory fragments from MemSearch-o1 and analysis sentences from Search-o1, together with retrieved document sentences, are projected into a low-dimensional space using UMAP (McInnes et al., 2018). Figure 4 shows two scenarios: (a) when both retrieval queries and retrieved regions differ, and (b) when they are identical.

As shown in Figure 4(a), memory growth allows MemSearch-o1 to generate queries that are more goal-directed and logically coherent, retrieving regions closer to the ground-truth answers. The richer semantics of memory fragments also enable broader exploration within the memory space, improving answer accuracy. In contrast, Figure 4(b) shows that even with identical retrieved regions, MemSearch-o1 outperforms Search-o1 (Refined). While Search-o1 often over-refines and restricts exploration to local information, MemSearch-o1



(a) UMAP: Memory Path under Divergent Retrieval Regions



(b) UMAP: Memory Path under Identical Retrieval Regions

Figure 4: Memory Paths for Different Retrieval Cases.

expands memory into a wider region, supporting more effective reasoning and retrieval. These visualizations clearly demonstrate how memory growth in MemSearch-o1 guides reasoning and retrieval in a more expansive, coherent, and accurate manner.

4 Related Work

Retrieval-Augmented Generation RAG (Lewis et al., 2020) has been a widely adopted method for enhancing LLMs with external knowledge (Jia et al., 2024). Single-step retrieval methods such as GraphRAG (Edge et al., 2024), HippoRAG (Gutiérrez et al., 2025), and HiRAG (Huang et al., 2025) leverage knowledge graphs or hierarchical structures to better exploit retrieved contexts. However, their limited retrieval scope often fails on complex multi-hop reasoning tasks (Qin et al., 2025). To address this, deep search has been proposed as an agentic RAG framework that integrates reasoning, planning, and retrieval (Li et al., 2025; El-Shorbagy et al., 2025; Zhang et al., 2025a), enabling iterative exploration. Yet, existing methods frequently generate redundant documents and noisy reasoning trajectories, leading to memory dilution and reduced effectiveness (Liu et al., 2023). In this work, we tackle this issue by introducing memory growth and retracing to construct compact, explicit memories that support high-quality reasoning. Unlike R1-style methods (Jin et al., 2025; Yu et al., 2025; Luo et al., 2025; Hao et al., 2025; Shi et al., 2025) based on smaller LLMs, our focus is on fully activating the reasoning potential of large-scale models.

Memory Management To mitigate memory dilution, prior work has explored a range of memory management strategies for LLMs. Search-o1 (Li et al., 2025) refines retrieved texts to retain the most useful information, while MemoryBank (Zhong et al., 2024) summarizes event histories and applies selective forgetting. Mem0 (Chhikara et al., 2025) combines multi-step retrieval with summarization to enrich semantic coverage, and Amber (Qin et al., 2025) filters relevant content at both chunk and sentence levels during memory updates. Despite these advances, such methods largely rely on summarization or retrieval of query-relevant texts, leaving LLMs unable to fully capture the complex semantic associations between evolving queries and retrieved contexts. This significantly constrains their deep search potential. In this work, we address this limitation by constructing concise yet semantically rich system memories, thereby unlocking more scalable deep search capabilities for LLMs.

5 Conclusion

In this paper, we propose MemSearch-o1, a novel deep search system that grows memory fragments from seeds and seeks a path, enhancing the deep search reasoning process, which significantly relieves the problem of memory dilution. This enables the deep search system to effectively extract concise information with rich semantics, and explore the memory path where the LLMs can search for and generate better solutions. Through extensive experiments on eight benchmark datasets, we demonstrate the effectiveness of MemSearch-o1 in enhancing the agentic search capability of LLMS, surpassing all the baselines, including RAG and agentic search frameworks with memory management. Our MemSearch-o1 aligned with reasoning based on memory growth establishes a new foundation for memory-aware agentic intelligence.

Limitations

While MemSearch-o1 effectively grows memory fragments and retraces memory paths for reasoning, it still depends on the comprehension ability of LRMs for memory seed preparation and growth. Our method demonstrates substantial improvements with large-parameter LLMs, but the gains on smaller models (fewer than 3B parameters) remain limited, as these models lack strong reasoning and information extraction capabilities and often fail to follow search instructions reliably. Enhancing the performance of small-scale models thus remains an important avenue for future work.

References

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, and 1 others. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Roger W Brown. 1957. Linguistic determinism and the part of speech. *The Journal of Abnormal and Social Psychology*, 55(1):1.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*.

David Maxwell Chickering. 2002. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*.

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

Mohammed A El-Shorbagy, Anas Bouaouda, Laith Abualigah, and Fatma A Hashim. 2025. Atom search optimization: a comprehensive review of its variants, applications, and future directions. *PeerJ Computer Science*, 11:e2722.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From rag to memory: Non-parametric continual learning for large language models. *arXiv preprint arXiv:2502.14802*.

Chuzhan Hao, Wenfeng Feng, Yuewei Zhang, and Hao Wang. 2025. Dynasearcher: Dynamic knowledge graph augmented search agent via multi-reward reinforcement learning. *arXiv preprint arXiv:2507.17365*.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, and 1 others. 2017. Dureader: a chinese machine reading comprehension dataset from real-world applications. *arXiv preprint arXiv:1711.05073*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Haoyu Huang, Yongfeng Huang, Junjie Yang, Zhenyu Pan, Yongqiang Chen, Kaili Ma, Hongzhi Chen, and James Cheng. 2025. Retrieval-augmented generation with hierarchical knowledge. *arXiv preprint arXiv:2503.10150*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Pengyue Jia, Derong Xu, Xiaopeng Li, Zhaocheng Du, Xiangyang Li, Yichao Wang, Yuhao Wang, Qidong Liu, Maolin Wang, Huifeng Guo, and 1 others. 2024. Bridging relevance and reasoning: Rationale distillation in retrieval-augmented generation. *arXiv preprint arXiv:2412.08519*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Seokgi Lee. 2025. Transforming questions and documents for semantically aligned retrieval-augmented generation. *arXiv preprint arXiv:2508.09755*.

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

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

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

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

Haoran Luo, Guanting Chen, Qika Lin, Yikai Guo, Fangzhi Xu, Zemin Kuang, Meina Song, Xiaobao Wu, Yifan Zhu, Luu Anh Tuan, and 1 others. 2025. Graph-r1: Towards agentic graphrag framework via end-to-end reinforcement learning. *arXiv preprint arXiv:2507.21892*.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Qitao Qin, Yucong Luo, Yihang Lu, Zhibo Chu, and Xianwei Meng. 2025. Towards adaptive memory-based optimization for enhanced retrieval-augmented generation. *arXiv preprint arXiv:2504.05312*.

Kaize Shi, Xueyao Sun, Qing Li, and Guandong Xu. 2024. Compressing long context for enhancing rag with amr-based concept distillation. *arXiv preprint arXiv:2405.03085*.

Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. 2025. Search and refine during think: Autonomous retrieval-augmented reasoning of llms. *arXiv preprint arXiv:2505.11277*.

Zoltán Gendler Szabó. 2015. Major parts of speech. *Erkenntnis*, 80(Suppl 1):3–29.

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.

Jun Wang. 2025. A tutorial on llm reasoning: Relevant methods behind chatgpt o1. *arXiv preprint arXiv:2502.10867*.

Yu Wang, Dmitry Krotov, Yuanzhe Hu, Yifan Gao, Wangchunshu Zhou, Julian McAuley, Dan Gutfreund, Rogerio Feris, and Zexue He. 2025a. M+: Extending memoryllm with scalable long-term memory. *arXiv preprint arXiv:2502.00592*.

Zihan Wang, Zihan Liang, Zhou Shao, Yufei Ma, Huangyu Dai, Ben Chen, Lingtao Mao, Chenyi Lei, Yuqing Ding, and Han Li. 2025b. Infogain-rag: Boosting retrieval-augmented generation via document information gain-based reranking and filtering. *arXiv preprint arXiv:2509.12765*.

Yaxiong Wu, Sheng Liang, Chen Zhang, Yichao Wang, Yongyue Zhang, Huifeng Guo, Ruiming Tang, and Yong Liu. 2025. From human memory to ai memory: A survey on memory mechanisms in the era of llms. *arXiv preprint arXiv:2504.15965*.

Wujiang Xu, Kai Mei, Hang Gao, Juntao Tan, Zujie Liang, and Yongfeng Zhang. 2025. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Hinrich Schütze, Volker Tresp, and Yunpu Ma. 2025. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

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 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Chuanyue Yu, Kuo Zhao, Yuhan Li, Heng Chang, Mingjian Feng, Xiangzhe Jiang, Yufei Sun, Jia Li, Yuzhi Zhang, Jianxin Li, and 1 others. 2025. Graphrag-r1: Graph retrieval-augmented generation with process-constrained reinforcement learning. *arXiv preprint arXiv:2507.23581*.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, and 1 others. 2025a. Process vs. outcome reward: Which is better for agentic rag reinforcement learning. *arXiv preprint arXiv:2505.14069*.

Yingyi Zhang, Pengyue Jia, Xianneng Li, Derong Xu, Maolin Wang, Yichao Wang, Zhaocheng Du, Huifeng Guo, Yong Liu, Ruiming Tang, and 1 others. 2025b. Lsrp: A leader-subordinate retrieval framework for privacy-preserving cloud-device collaboration. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 3889–3900.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19724–19731.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. 2025. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*.

A Appendix

A.1 Algorithm

The detailed inference procedure of MemSearch-o1 is shown in Algorithm 1. For MemSearch-o1 inference, we should prepare the model with reasoning ability $\mathcal{R}$, search function `Search`, toolkit for memory seed preparation $\mathcal{T}$. We input the original query, and instructions for reasoning, memory growth and answers to the LLM to execute inference. Before the inference starts, we initialize the maximum search round as N, and the current query $\boldsymbol{q}$ as the original $\boldsymbol{q}_o$ for first-step analysis. While the current search round is less than N, and the search indicator is True, the LLM should generate a sequence of analysis using current search query. In line2, if the reasoning sequence of the current round $\boldsymbol{R}^{(n)}$ ends with ⟨|end_search_query|⟩), this means the model $\mathcal{R}$ should execute search to seek for more information. Then, the raised query $\boldsymbol{q}^{(n)}$ is extracted from $\boldsymbol{R}^{(n)}$, and is utilized to retrieve relevant contexts in line 5. In line 6, we use the toolkit to split the query and prepare for memory seed tokens which is stored in the set $\mathcal{S}^{(n)}$. In line 7, we grow memory fragments using the LLM based on the intent of current query $\boldsymbol{q}^{(n)}$. If the reasoning sequence $\boldsymbol{R}^{(n)}$ ends with EOS, the model regards the retrieved information as sufficient, and finishes the reasoning process. Finally, we should consolidate the memory fragments and calculate the contribution function to find a memory region in line 11 and 12. In line 13, we use greedy search to deeply refine and reorganize the memory path for reasoning, as the some memory fragments might have the semantics far away from that of the original query. Finally in line 14, we use the model $\mathcal{R}$ to generate the answer a.

Task	Metric	Avg. Length	Language	#Samples
HotpotQA	F1	9,151	EN	200
2WikiMultihopQA	F1	4,887	EN	200
MuSiQue	F1	11,214	EN	200
DuReader	Rouge-L	15,768	ZH	200
MultiFieldQA-en	F1	4,559	EN	150
MultiFieldQA-zh	F1	6,701	ZH	200
NarrativeQA	F1	18,409	EN	200
Qasper	F1	3,619	EN	200

Table 3: Statistics of Evaluation Datasets

A.2 Details for the Datasets

In our experiments, we utilize LongBench (Bai et al., 2023) to evaluate the performance of methods, which covers both Chinese and English tasks, offering a more complete assessment of how well models handle questions and comprehend information in long contexts in different languages. We select MultiDocQA and SingleDocQA tasks for evaluation, and the brief description of adopted domains are listed as below:

HotpotQA (Yang et al., 2018): HotpotQA is a question-answering dataset that includes natural, multi-step questions. It provides clear labels for the supporting facts, helping build question-answering systems that are easier to explain.

2WikiMQA (Ho et al., 2020): 2WikiMQA is a high-quality multi-hop QA dataset designed to evaluate a model's reasoning and inference capabilities. The dataset incorporates evidences by combining structured and unstructured data, and explicitly provides reasoning paths for multi-hop questions.

MuSiQue (Trivedi et al., 2022): MuSiQue is a challenging multi-hop QA dataset constructed via a bottom-up approach that composes connected single-hop questions, ensuring that each reasoning step depends on information from another.

DuReader (He et al., 2017): DuReader is a human-annotated Chinese machine reading comprehension dataset grounded in real-world QA tasks. It emphasizes authenticity by featuring genuine user queries, naturally occurring documents, human-provided answers, and practical application contexts.

NarrativeQA (Kočiskỳ et al., 2018): NarrativeQA is a reading comprehension dataset designed to evaluate deep, integrative understanding of long-form narratives, such as books and movie scripts.

Qasper (Dasigi et al., 2021): Qasper is a QA dataset specifically designed for NLP papers. Questions are authored by NLP practitioners who only

Algorithm 1 MemSearch-o1 Inference

Require: LLM for reasoning $\mathcal{R}$, search function `Search`, toolkit for memory seed preparation $\mathcal{T}$.
Input: Original query $\boldsymbol{q}_o$, task instruction $\boldsymbol{I}_R$, answer instruction $\boldsymbol{I}_a$, memory growth instruction $\boldsymbol{I}_M$
Initialize the search indicator $\mathcal{F} =$ True, Corpus $\mathcal{C}$ for retrieval and maximum round of search N.
Initialize the memory seed set $\mathcal{S}$ and memory fragment set $\mathcal{M}$ as empty, and set search round $n = 1$.
Initialize the current query $\boldsymbol{q}$ for analysis and retrieval as $\boldsymbol{q}_o$, and the reasoning text $\boldsymbol{R}^{(0)}$ as empty.

while $n \leq N$ and $\mathcal{F} =$ True **do**
  Generate reasoning and derive an analysis sequence. $\boldsymbol{R}^{(n)} \leftarrow \mathcal{R}(\boldsymbol{I}_R, \boldsymbol{q}, \boldsymbol{R}^{(n-1)})$
  if $\boldsymbol{R}^{(n)}$ ends with ⟨|end_search_query|⟩ **then**
    Extract search query: $\boldsymbol{q}^{(n)} \leftarrow$ Extract($\boldsymbol{R}^{(n)}$, ⟨|begin_search_query|⟩, ⟨|end_search_query|⟩)
    Retrieve documents: $\boldsymbol{D} \leftarrow \texttt{Search}(\boldsymbol{q}^{(n)})$
    Prepare memory seed tokens $\mathcal{S}^{(n)} = \{\boldsymbol{s}_1^{(n)}, \boldsymbol{s}_2^{(n)}, \ldots, \boldsymbol{s}_{L_r}^{(n)}\} \leftarrow \mathcal{T}(\boldsymbol{q}^{(n)})$
    Grow memory fragments $\mathcal{M}^{(n)} = \{\boldsymbol{m}_1^{(n)}, \boldsymbol{m}_2^{(n)}, \ldots, \boldsymbol{m}_{|\mathcal{M}^{(n)}|}^{(n)}\} \leftarrow \mathcal{R}(\mathcal{S}^{(n)}, \boldsymbol{I}_M, \boldsymbol{q}^{(n)})$.
    $n = n + 1$
  else if $\boldsymbol{R}^{(n)}$ ends with EOS **then**
    Finish the Reasoning Step. $\mathcal{F} =$ False
    Memory consolidation: $\mathcal{M} = \{\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_{|\mathcal{M}|}\} \leftarrow \bigcup_{n=1}^{N} \mathcal{M}^{(n)}$
    Calculate the contribution to find memory region: $\mathcal{M}_q = \{\boldsymbol{m_i} \in \mathbb{R}^{\boldsymbol{d}} | C(\boldsymbol{m}_i; \boldsymbol{q}_o) > \tau_r\}$.
    Find an optimal memory path $\mathcal{P}^* = (\boldsymbol{m}_{j_1}, \boldsymbol{m}_{j_2}, ..., \boldsymbol{m}_{j_K})$ using greedy search.
    Generate the final response: $\boldsymbol{a} \leftarrow \mathcal{R}(I_a, \boldsymbol{q}_o, \mathcal{P}^*)$
  end if
end while

Output: High quality reasoning chain $\boldsymbol{R}$ and answer $\boldsymbol{a}$

read the title and abstract, yet seek information that resides in the full paper; answers and supporting evidence are then provided by other experts.

MultiFieldQA (Bai et al., 2023): MultiFieldQA is a manually curated QA dataset designed to evaluate long-context comprehension across diverse domains, with versions in English and Chinese.It draws documents from legal texts, government reports, encyclopedias, and academic papers.

The statistics of the above domains for evaluation are shown in Table 3. The above statistics show that our evaluation is conducted on a diverse set of datasets, ranging from those with shorter contexts where query-relevant information is easier to retrieve—to those with much longer contexts, posing greater challenges for information retrieval. MultiDocQA tasks require synthesizing evidence across multiple, often disjoint, passages to answer complex, multi-hop questions, while SingleDoc QA focuses on precise comprehension and reasoning within a single, typically longer, document.

A.3 Baselines

In this subsection, we will introduce the baselines adopted in our experiments in detail.

Direct RAG (Lewis et al., 2020): Retrieve relevant documents as external knowledge from the corpus to enhance the the generation quality of LLMs.

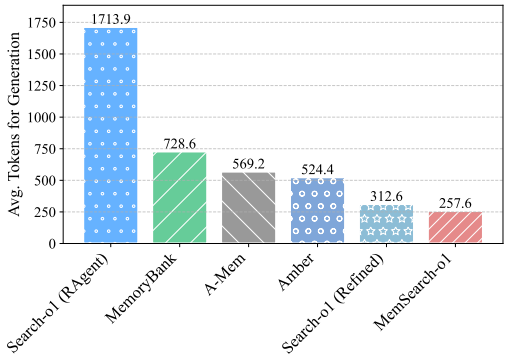
Search-o1 (RAgent) (Li et al., 2025): Search-o1 as a pioneering deep search system, enables LLMs to integrate agentic search into o1-like reasoning process, during which the retrieved documents are used for inference directly.

Search-o1 (Doc Reasoning) (Li et al., 2025): This is a deep search system integrating memory refinement into reasoning. It summarizes the retrieved contexts to improve the quality of reasoning.

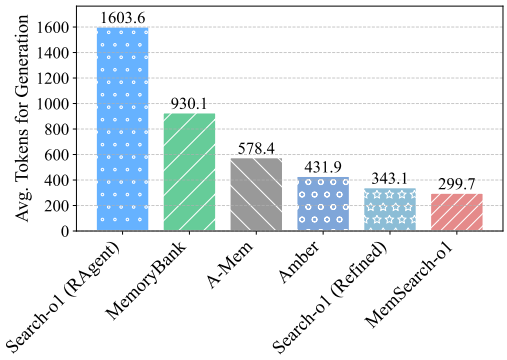
MemoryBank (Zhong et al., 2024): A typical memory management method, and is applied to deep search system in our experiment. It summarizes the key events from the retrieved texts, and employs forgetting mechanism properly.

A-Mem (Xu et al., 2025): A novel memory management method, and is applied to deep search system in our experiments. It generates key notes for retrieved documents, and evolve the memories using links between the notes.

Amber (Qin et al., 2025): A deep search system using a novel system memory management method, which filters the retrieved documents in chunk and



(a) HotpotQA.



(b) 2WikiMQA.

Figure 5: Average Tokens for Generation.

sentence levels to optimizes the reasoning process.

A.4 Average Tokens for Generation

In this subsection, we show the average tokens for generating answers across all baseline methods. The experiments are conducted on HotpotQA and 2WikiMQA, using the DeepSeek-V3.1 backbone. We limit the maximum number of search iterations to three. The results are displayed in Figure 5.

The experimental results show that refining retrieved texts without careful processing leads to verbose memory representations, causing severe memory dilution and thereby increasing the difficulty for LLMs to generate accurate responses. This finding is consistent with the instability of Search-o1 (RAgent) observed in Table 1. Directly applying the document refinement strategy from Search-o1, however, tends to over-summarize the content, resulting in significant information loss, as such refinement often overlooks the complex semantics of the queries generated during deep search. Although other memory refinement approaches, such as A-Mem and Amber, achieve relatively better refinement performance, they still extract irrelevant information from documents, ultimately degrading generation quality. Uniquely, our MemSearch-o1 method eliminates the reliance on reasoning history inherent in traditional deep search frameworks. Instead, it grows memory structures from memory seed tokens in a targeted way and then identifies optimal memory paths from the grown memory fragments for response generation. This approach substantially reduces the number of tokens required during generation, improving both efficiency and generation quality while effectively mitigating the memory dilution problem.

top-k	2WikiMQA		MuSiQue	
	F1	Thinking rounds	F1	Thinking rounds
1	51.14	3.43	30.26	3.66
2	59.94	3.29	33.54	3.35
3	65.95	3.13	41.57	3.37
5	67.45	2.99	42.94	3.38
10	65.85	3.05	44.05	3.20

Table 4: Performance and Reasoning Cost vs. Top-k Retrieval on 2WikiMQA and MuSiQue

A.5 Detailed Values of Top-k Scalability

In this subsection, we supplement the detailed values of the top-k scalability, and analyze the mechanism behind the observed results.

As shown in Table 4, as k increases, a richer set of information sources will be retrieved for memory construction. Consequently, during the memory seed expansion process, the growing memory fragments benefit from more contextual input, resulting in semantically richer representations. As a result, the deep search performance improves rapidly in the initial phase of increasing k. Moreover, the higher-quality retrieved information boosts the model's confidence, leading to a reduction in the number of search iterations. This phenomenon is especially obvious on the results of MuSiQue dataset. However, as the retrieved information becomes extremely large, redundant texts will be fed into the LLMs. Since the memory fragments growth are based on both memory seeds and the retrieved texts, too lengthy texts may also lead to the context dilution, and guide the memory growth to irrelevant information region.

A.6 Prompts

The instructions for agentic search follow the previous work (Li et al., 2025; Wang et al., 2025a). We take the HotpotQA as an example, and the instructions for deep search are shown in Table 5. For the memory fragments growth, we organize the prompt template in Table 6. The prompts for answer generation is shown in Table 7.

You are a reasoning assistant with the ability to perform searches to help you answer the user's question accurately. **When answering, just give the answer and do not output other information.**
You have special tools:
- To perform a search: write <|begin_search_query|> your query here <|end_search_query|>.
Then, the system will search and analyze relevant passages, then provide you with helpful information in the format <|begin_search_result|> ...search results... <|end_search_result|>.
If you think the searched information is not enough, you can continue searching. The maximum number of search attempts is limited to {MAX_SEARCH_LIMIT}.
Once you have all the information you need, stop the search and continue your reasoning.
Example:
Question: "Alice David is the voice of Lara Croft in a video game developed by which company?"
Assistant thinking steps:
- I need to find out who voices Lara Croft in the video game.
- Then, I need to determine which company developed that video game.
Assistant:
<|begin_search_query|>Alice David Lara Croft voice<|end_search_query|>
(System returns processed information from relevant passages)
Assistant thinks: The search results indicate that Alice David is the voice of Lara Croft in a specific video game. Now, I need to find out which company developed that game.
Assistant:
<|begin_search_query|>video game developed by Alice David Lara Croft<|end_search_query|>
(System returns processed information from relevant passages)
Assistant continues reasoning with the new information...
Remember:
- Use <|begin_search_query|> to request a search and end with <|end_search_query|>.
- When done searching, continue your reasoning.

Table 5: Prompt Used for the Reasoning Assistant

Please extract content from the provided text that is useful for answering the given query, specifically with respect to the listed subjects, actions, temporal markers, degree modifiers.
Input:
- List of subjects, actions, temporal markers, degree modifiers:
- Text: ...
- Query: ...
Instructions:
- For each item in the list of subjects, actions, temporal markers or degree descriptions, extract raw, verbatim content from the text that is directly relevant to the query.
- Format each extracted piece as:
  `subjects: [exact content from the text about the subjects]`
  `actions: [exact content from the text describing an action involving the verb]`
  `temporal markers: [exact content from the text indicating when an event occurred or the time duration]`
  `degree modifiers: [exact content from the text indicating the characteristics of the actions and subjects.]`
- Do not paraphrase, summarize, or use your own words. Use only direct excerpts or minimally truncated phrases that preserve original wording.
- Each subjects/actions/temporal markers/degree modifiers–content pair must appear on a separate line.
- Only include content that provides diverse and query-relevant information.
- If no relevant content exists in the text for a given entity, verb, or time expression with respect to the query, omit it entirely.

Table 6: Prompt for Memory Fragments Growth

Answer the question based on the given system memory of reasoning. Just give the answer and do not output other information.
You should provide your final answer in the format \boxed{YOUR_ANSWER}.
If the answer is in the context, maintain the illustrations (e.g., examples and specific phrasings) present in the context when formulating the answer.
System memory: ...
Question:...
Generate an accurate answer based solely on the provided information.

Table 7: Prompt for Answering Questions Based on Given Context