Scaling LLM Agents with Self-Evolving Structured Memory

Anonymous ACL submission

Abstract

With the growing adoption of large language model (LLM) agents in persistent real-world roles, they increasingly face continuous streams of complex tasks requiring iterative reasoning and evidence integration. A common strategy for improving agent performance is inference-time scaling, which allocates additional computation to exploration and reasoning. However, existing self-evolving agents typically scale by expanding interaction trajectories, leaving the knowledge accumulated largely unstructured and making it difficult to consolidate discoveries or guide further reasoning. We address this limitation by reframing inference-time scaling as structured knowledge accumulation rather than trajectory expansion. We propose StructMem, a framework that represents the agent's evolving knowledge state as a dynamically constructed knowledge graph. During exploration, newly discovered facts are incrementally integrated into the graph, where structural constraints enable conflict detection, knowledge gap identification, and consistency-aware reasoning. Through this iterative build–verify–expand process, StructMem progressively constructs a coherent knowledge structure that supports grounded and verifiable answer generation. Experiments on challenging knowledge-intensive benchmarks demonstrate substantial improvements over existing agents in reasoning accuracy and computational efficiency. The code is available at `https://anonymous.4open.science/r/StructMem-code-0104`.

1 Introduction

Large language models (LLMs) are increasingly evolving from standalone question-answering systems into autonomous agents capable of performing complex, long-horizon tasks (Liu et al., 2025; Wang et al., 2024). In open-world environments, these agents must iteratively gather information, consolidate new knowledge, and guide subsequent reasoning. However, without effective organization of intermediate discoveries, long interaction histories can become noisy and hard to exploit (Gur et al., 2023; Shi et al., 2025a). Recent work has explored inference-time scaling to improve performance by allocating more computation to reasoning and exploration (Snell et al., 2025), including expanding interaction trajectories or exploring multiple reasoning paths (Zheng et al., 2023b; Kagaya et al., 2024).

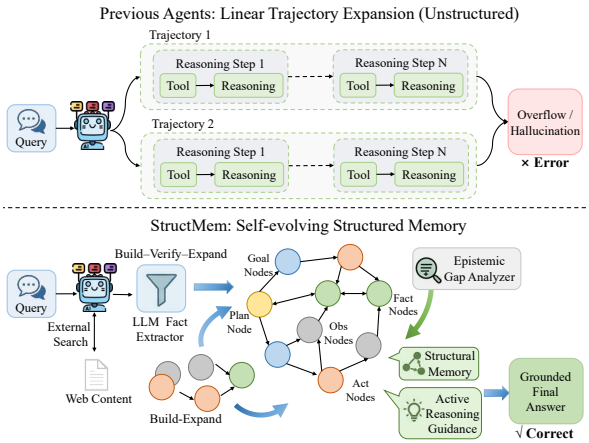

Figure 1: Comparison of inference-time scaling strategies for LLM agents: linear trajectory expansion versus the self-evolving structured memory in StructMem.

Limitations of trajectory-based memory. In these systems, discovered knowledge is typically stored as linear interaction histories or simple memory buffers (Fang et al., 2025), where new information remains embedded in long reasoning traces rather than being explicitly consolidated. This design has two key limitations in effectiveness and efficiency. (i) *Effectiveness limitation: lack of explicit reasoning guidance.* Trajectory-based memory buffers primarily serve as passive records rather than structured representations of knowledge. Consequently, they cannot explicitly represent relationships among retrieved facts, detect logical inconsistencies, or reveal missing information required to solve the task. Without such structural signals, the

agent lacks a clear understanding of what knowledge has been established and what evidence is still required. (ii) *Efficiency limitation: lack of effective information refinement.* As agents continuously append raw observations to their interaction histories, the accumulated context grows rapidly and contains substantial redundancy. Without distilling observations into compact structured facts, the model must repeatedly process large volumes of unstructured text at every reasoning step. This leads to significant computational overhead and poor scalability as the interaction trajectory becomes longer.

Structured memory instead of passive trajectories. Insights from cognitive science suggest that human reasoning rarely relies on passive sequential logs of observations (Bieth et al., 2024). Instead, humans construct structured semantic representations that organize knowledge into interconnected relations, enabling them to detect inconsistencies, infer missing information, and guide subsequent reasoning. This observation suggests that effective long-horizon agent reasoning requires memory mechanisms that go beyond flat textual histories. As illustrated in Figure 1, rather than passively recording interaction trajectories, an agent's working memory should actively organize discovered knowledge into a structured representation that evolves during exploration.

Self-evolving structured memory for LLM agents. Motivated by this perspective, we propose StructMem, a framework that scales LLM agents through self-evolving structured memory. Instead of expanding unstructured interaction trajectories, StructMem represents the agent's evolving epistemic state as a dynamically constructed knowledge graph.

During reasoning, the framework performs a continuous build–verify–expand cycle. As the agent interacts with external environments, raw observations are distilled into factual triples by an LLM-based extractor. These facts are integrated into the existing knowledge graph via a semantic relational integrator, which merges redundant information and identifies contradictions. An epistemic gap analyzer evaluates the evolving graph with respect to the root task, detecting missing knowledge and generating targeted exploration queries to guide subsequent reasoning steps. Through this iterative process, the agent progressively constructs a coherent knowledge structure, enabling more informed exploration and grounded answer generation.

Experimental results. We evaluate StructMem on four challenging deep-search benchmarks: BrowseComp (Wei et al., 2025a), BrowseComp-ZH (Zhou et al., 2025), GAIA (Mialon et al., 2023), and xbench-DeepSearch (Chen et al., 2025). Experimental results show that StructMem consistently outperforms advanced structured agents and proprietary deep-search systems, achieving a 19.5% improvement over OpenAI DeepResearch (OpenAI, 2025) on BrowseComp. Further experiments demonstrate that StructMem improves execution efficiency and generalizes well to specialized domains such as MedXpertQA (Zuo et al., 2025) without task-specific tuning.

2 Related Work

Deep search agents. Driven by the advanced capabilities of LLMs, recent research has developed deep search agents to tackle complex, multi-step information-seeking tasks (Nakano et al., 2021; Trivedi et al., 2023; Jin et al., 2025; Wei et al., 2025b). Unlike basic, single-turn question-answering models, these agents operate in open environments. They use iterative loops of reasoning and tool execution to continuously search the web, read pages, and gather evidence (Yao et al., 2022; Shinn et al., 2023; Qin et al., 2025; Shi et al., 2025b). The progress in this field is strongly supported by challenging benchmarks like BrowseComp (Wei et al., 2025a) and GAIA (Mialon et al., 2023), which require agents to persistently navigate the internet and synthesize information. To succeed on these tasks, modern agents typically maintain a text-based history of past actions and observations to plan their future steps (Wang et al., 2024; Ouyang et al., 2025).

Trajectory-based memories face inherent limitations. As interaction histories grow longer, early information may be forgotten, leading to unstable behavior such as repeated queries or loss of focus on the original goal (Liu et al., 2024). Addressing these limitations motivates methods that can organize knowledge more explicitly and guide exploration more effectively.

LLMs for graph reasoning. To enhance reasoning and retrieval, many works integrate LLMs with graph structures. Graph-based retrieval-augmented generation leverages external knowledge graphs to improve multi-hop question answering (Edge et al., 2024; Yasunaga et al., 2022; Zhu et al.,

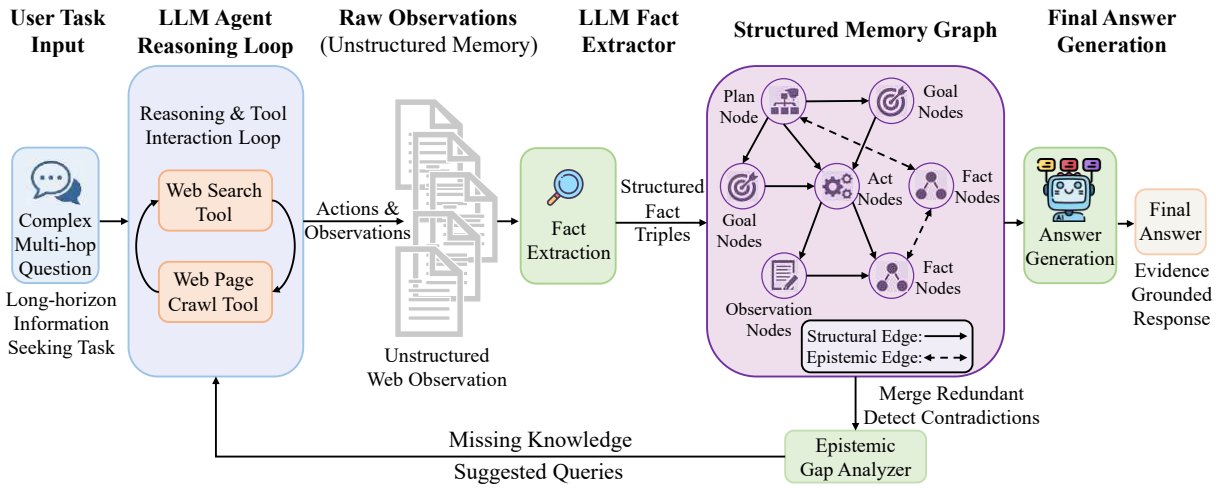

Figure 2: The StructMem framework for scaling LLM agents. Structured facts are extracted from raw observations, incorporated into a self-evolving knowledge graph, and analyzed for epistemic gaps, enabling guided exploration and graph-grounded answer synthesis.

2025). Other approaches, e.g., Tree of Thoughts (Yao et al., 2023) and Graph of Thoughts (Besta et al., 2024), structure the LLM's internal reasoning process, decomposing complex problems into connected steps and enabling exploration of multiple reasoning paths. Flash-Searcher (Qin et al., 2025) introduces a reasoning architecture based on directed acyclic graphs that enable parallel execution of subtasks and improves accuracy. However, its inference still relies on parallel trajectory expansion, where intermediate knowledge is not explicitly structured and accumulated, leading to redundant exploration and limited reuse of discovered information. This limitation calls for approaches that dynamically construct and evolve structured knowledge during reasoning.

3 Method

To address the efficiency and effectiveness limitations of trajectory-based memory, we propose StructMem. It integrates a dynamically evolving structured memory directly into the agent's reasoning loop. StructMem models the agent's epistemic state as a knowledge graph that evolves with exploration, via an iterative build–verify–expand cycle (Figure 2) that has three key components: (i) the definition of structural memory elements (§3.1); (ii) dynamic memory evolution via the semantic relational integrator and epistemic gap analyzer (§3.2); and (iii) graph-grounded answer generation based on the consolidated memory (§3.3).

3.1 Structural memory: definition

We represent the agent's memory as a directed graph $\mathcal{G}_t = (\mathcal{V}, \mathcal{E})$, dynamically built during task execution. Unlike static knowledge bases, this memory is open-domain and continuously updated by the LLM.

Node types. Nodes represent task-relevant information at varying levels of abstraction: $\mathcal{V} = \{v_{plan}, v_{goal}, v_{action}, v_{obs}, v_{fact}\}$, where

- v_{plan}: high-level strategy parsed from the task.
- v_{goal}: intermediate sub-goals decomposed from the plan.
- v_{action}: executed tool operations (e.g., web search).
- v_{obs}: raw textual feedback retrieved from the environment.
- v_{fact}: structured facts extracted from observations.

To ensure expressiveness, each v_{fact} node is represented as an augmented RDF triples (Klyne, 2004): $\mathcal{F} = \{(e_s, r, e_o, c)\}$. Here, e_s is the subject, r is the relation, e_o is the object, and c is an importance score (e.g., critical or supplementary) assigned by the LLM. The score c helps the system prioritize information when generating the answer.

Edge types. The edge set $\mathcal{E}$ defines the structural and semantic connectivity between nodes. It consists of two main categories: structural dependencies and epistemic relationships.

Structural dependencies track the agent's execution flow, where (i) the Contains edge represents general inclusion, such as linking a high-level plan to its specific sub-goals; and (ii) the Produces edge connects a tool action to its resulting raw observation. *Epistemic relationships* map the agreement between different extracted facts, where (i) the Supports edge indicates corroborating evi-

dence; (ii) the CONTRADICTS edge highlights conflicting information; and (iii) the RELATED edge denotes general semantic relevance.

3.2 Structural memory: evolving

The memory graph $\mathcal{G}_t$ evolves iteratively from an initial task decomposition, integrating new observations while maintaining global consistency and highlighting knowledge gaps to guide exploration.

Initialization. Given a user task T, an LLM parser decomposes it into a high-level plan and sub-goals, forming the base graph $\mathcal{G}_0$. The central v_{plan} node connects to v_{goal} nodes (CONTAINS edges). Task-provided facts are linked to v_{plan}, grounding initial knowledge.

Continuous expansion. At each reasoning step t, a tool action a_t (e.g., a web search or page crawl) produces unstructured observation o_t, prompting a graph update from $\mathcal{G}_t$ to $\mathcal{G}_{t+1}$. A new v_{action} node represents a_t and links to the governing v_{plan}. The resulting observation o_t is stored in a v_{obs} node connected via a PRODUCES edge. An LLM-based fact extractor $f_{ext}(o_t, T)$ converts o_t into candidate factual triples $\mathcal{F}_t$, which are instantiated as v_{fact} nodes linked to v_{obs} through CONTAINS edges. This process converts unstructured raw data into structured knowledge while preserving causal relationships.

Fact integration via semantic relational integrator. Before appending new facts to the global graph, a semantic relational integrator evaluates the extracted fact $\mathcal{F}_{new}$ against each existing fact $\mathcal{F}_{exist}$. Specifically, the LLM computes a pairwise relation $R(\mathcal{F}_{new}, \mathcal{F}_{exist}) \in \{\text{SUPPORTS}, \text{CONTRADICTS}, \text{RELATED}, \text{NONE}\}$. This mechanism actively maintains the memory's purity: if $\mathcal{F}_{new}$ is semantically identical to an existing fact $\mathcal{F}_{exist}$, the nodes are merged to prevent redundant context bloat. Conversely, if they conflict, a CONTRADICTS edge is formed. This explicitly highlights an epistemic discrepancy within the memory, signaling the agent to resolve the contradiction in subsequent exploration steps.

Reasoning guidance via epistemic gap analyzer. To prevent the agent from executing redundant queries or failing to recognize task completion, we introduce an epistemic gap analyzer. The analyzer evaluates the current knowledge against the root plan v_{plan}. Formally, the gap analyzer acts as a deductive function $H(\mathcal{G}_t, T) \rightarrow (\mathcal{K}_{cov}, \mathcal{K}_{mis}, \mathcal{Q}_{sug})$:

- $\mathcal{K}_{cov}$: The subspace of the task covered by high-confidence factual sub-graphs.
- $\mathcal{K}_{mis}$: The identified structural information gaps (i.e., missing relationships needed to connect a v_{goal} to v_{fact} nodes).
- $\mathcal{Q}_{sug}$: A set of precise, dynamically synthesized queries targeting $\mathcal{K}_{mis}$.

The analysis is prepended to the LLM context, steering subsequent reasoning.

The agent iteratively performs actions, extracts observations, integrates facts, and updates epistemic gaps until either all structural gaps are resolved or a maximum step limit is reached, producing a complete, consistent knowledge graph.

3.3 STRUCTMEM-based answer generation

The agent generates its final response from the consolidated memory graph. Facts are organized by LLM-assigned importance scores, forming a hierarchical context. This structured knowledge, together with the graph's epistemic confidence, is fed into the final synthesis prompt. Consequently, the LLM acts as an answer generator that explicitly synthesizes its response based on the evidence. This mechanism ensures that all claims in the final answer are grounded in retrieved, structurally verified facts.

4 Experimental Settings

Benchmark datasets. To rigorously validate the information-seeking and reasoning capabilities of STRUCTMEM in complex, human-challenging scenarios, we evaluate our approach on a suite of demanding benchmarks rather than relatively simple question-answering datasets. (i) **BrowseComp** (Wei et al., 2025a): A web browsing benchmark with 1,266 complex tasks designed to evaluate an agent's ability to perform persistent web browsing and multi-step search to discover hard-to-find information. (ii) **BrowseComp-ZH** (Zhou et al., 2025): A Chinese web browsing benchmark with 289 multi-hop questions that evaluates information-seeking and multihop reasoning in the Chinese web environment. (iii) **GAIA** (Mialon et al., 2023): A benchmark for general AI assistants that tests real-world problem solving requiring reasoning, tool use, and web browsing. We use its text-only validation set. (iv) **xbench-DeepSearch** (Chen et al., 2025): A subset of the xbench evaluation suite that measures deep-search capability by evaluating search breadth and reasoning depth in complex tasks.

For BrowseComp, we follow (Zeng et al., 2026) and conduct our experiments on the same sampled subset of 100 examples, since the computational

costs of running experiments on the full dataset would be too high. The selected 100 examples in BrowseComp were previously subjected to robustness analyses in (Zeng et al., 2026) to ensure their fairness and representativeness. For all other datasets, we conduct evaluations on the full dataset.

Evaluation metrics. We adopt the LLM-as-Judge paradigm (Zheng et al., 2023a; Qin et al., 2025) for automated evaluation, using GPT-4.1-mini as the judge model. For each benchmark, the judge model provides a binary correctness decision for the output generated by the agent. In reporting overall performance, we present accuracy as the default metric.

Baseline models. We compare StructMem with existing methods from three categories. (i) **Proprietary deep research agents**: *Grok3 DeepResearch* (xAI, 2025a,b), *Doubao DeepResearch* (ByteDance Doubao, 2025), *Tongyi DeepResearch* (Tongyi DeepResearch Team, 2025), *OpenAI DeepResearch* (OpenAI, 2025) and *OpenAI ChatGPT agent*. (ii) **Advanced LLMs**: *DeepSeek-R1* (Guo et al., 2025), *Gemini-2.5-Pro* (Comanici et al., 2025), and *OpenAI GPT-5* (Singh et al., 2025). (iii) **Structured agents**: *MiroFlow* (MiroMind AI Team, 2025), *Flash-Searcher* (Qin et al., 2025), and *StructMem (Structured Document)*. While *MiroFlow* and *Flash-Searcher* structure the agent workflow, STRUCTMEM structures the accumulated knowledge itself.

To more directly show the effectiveness of our graph-based structured memory, we introduce *StructMem (Structured Document)*, which replaces the knowledge graph with a simpler structured-document memory while keeping other settings unchanged. Since Flash-Searcher does not report results on BrowseComp-ZH, we reproduce its performance using the official implementation under the same evaluation setup. Other model results are taken from official reports.

Implementation details. We use GPT-5 (Singh et al., 2025) as the base model. It acts as both the main reasoning engine and the manager for all knowledge graph updates. To save computing time and prevent the agent from getting stuck in endless loops, we limit the agent to a maximum of 40 steps per task. The gap analyzer is set to trigger periodically. To make the agent more flexible and avoid fixed patterns, the gap analysis step happens roughly every 8 steps, with slight random changes during the run.

To ensure efficient agent workflows, STRUCTMEM employs a two-tool configuration: (i) A search tool (Serper API; Serper, Inc., 2025): Returns the top 5 relevance-ranked results (title, snippet, URL) per query. This balances comprehensive information access with computational efficiency, preventing information overload. (ii) A crawl tool (Jina Reader; Inc. Jina, 2025): Extract and summarize the online content from a web page. To manage context limits and API costs, pages are truncated to the first 60,000 characters before summarization. This streamlined setup provides robust retrieval capabilities, making it highly effective for the framework's parallel execution architecture.

5 Experimental Results

We evaluate STRUCTMEM through six research questions: (i) (**RQ1**) Does structured memory improve reasoning over standalone LLMs and structured agents? (ii) (**RQ2**) How does the memory graph adapt to task complexity and compress information? (iii) (**RQ3**) Can STRUCTMEM improve execution efficiency by reducing LLM-side token consumption and redundant retrieval? (iv) (**RQ4**) How much does each key component contribute to the overall performance? (v) (**RQ5**) How sensitive is performance to the choice of backbone LLM? (vi) (**RQ6**) Can the framework generalize to specialized, knowledge-intensive domains?

5.1 RQ1: Main results

Table 1 compares STRUCTMEM with proprietary deep-research agents, advanced reasoning models, and structured agents across four challenging deep-search benchmarks.

Static models vs. search agents. We first observe a substantial performance gap between standalone reasoning models and agent-based systems. For example, GPT-4o and DeepSeek-R1 achieve only 0.6% and 2.0% accuracy on BrowseComp, respectively. Even the strongest standalone model (OpenAI GPT-5) achieves just 19.8%, which remains noticeably below most agent-based approaches. This trend highlights a key limitation of purely parametric reasoning: complex deep-search tasks require iterative information acquisition and evidence integration, which static models cannot perform.

Performance of existing agents. Agent frameworks significantly improve performance by enabling external search and iterative reasoning. Among the structured agents, Flash-Searcher achieves the strongest baseline results: 67.7% on

Model	Benchmark Accuracy (%)			
	BrowseComp	BrowseComp-ZH	GAIA	xbench-DeepSearch
Proprietary Agents				
Grok3 DeepResearch	-	12.9	-	50+
Doubao DeepResearch	-	26.0	-	50+
Tongyi DeepResearch	43.4	46.7	70.9	75.0
OpenAI DeepResearch	51.5	42.9	70.5	66.7
OpenAI ChatGPT agent	68.9	-	-	-
Models				
DeepSeek-R1	2.0	23.2	16.5	32.7
Gemini-2.5-Pro	7.6	27.3	-	-
OpenAI GPT-5	19.8	34.3	-	30.0
Structured Agents (GPT 5)				
StructMem (Structured Document)	59.0	54.7	64.1	70.0
MiroFlow	63.4	59.7	71.9	73.5
Flash-Searcher	67.7	60.2	82.5	83.0
STRUCTMEM	**71.0**	**66.8**	**83.5**	**87.0**

Table 1: Performance comparison with proprietary deep-research agents, advanced reasoning models, and structured agents on four challenging deep-search benchmarks.

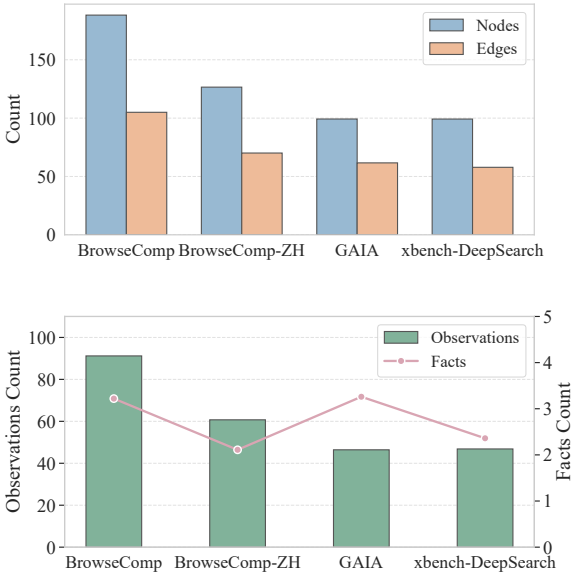

Figure 3: Structured memory statistics. Top: Average nodes vs. edges. Bottom: Raw observations vs. extracted facts.

BrowseComp and 83.0% on xbench-DeepSearch. Similarly, proprietary systems such as OpenAI DeepResearch demonstrate strong performance on several benchmarks. These results confirm that search-based reasoning pipelines are crucial for long-horizon information-seeking tasks.

Effectiveness of structured memory. STRUCTMEM consistently achieves the best results across all benchmarks. In particular, it reaches 71.0% on BrowseComp and 66.8% on BrowseComp-ZH, outperforming the strongest structured agent Flash-Searcher by +3.3 and +6.6 points respectively. On GAIA and xbench-DeepSearch, STRUCTMEM improves the best existing results to 83.5% and 87.0%, establishing new state-of-the-art performance. These improvements highlight the core advantage of STRUCTMEM. While existing agents primarily accumulate reasoning traces as linear trajectories, our framework explicitly organizes discovered information into a dynamically evolving knowledge graph. Moreover, STRUCTMEM outperforms StructMem (Structured Document) on all benchmarks, showing the advantage of graph-based structured memory over a document-style structure. This structured memory enables more consistent evidence integration and better reasoning guidance, leading to higher accuracy on complex deep-search tasks.

Case study. We provide a representative multi-hop reasoning case study from BrowseComp in Table 6 in Appendix A.1. The example shows that STRUCTMEM can effectively compress large-scale noisy web observations into concise structured knowledge for focused multi-hop reasoning, while also revealing remaining challenges in robustness to noisy fact extraction.

5.2 RQ2: Structured memory analysis

To examine the internal behavior of STRUCTMEM, we collected statistics from the knowledge graphs produced by the agent across tasks in four benchmarks. Figure 3 reports the average number of nodes, edges, v_{obs} nodes, and v_{fact} nodes per task.

Adaptive expansion. As shown in the top panel of Figure 3, on the BrowseComp dataset, the most complex benchmark focusing on deep web exploration, the graph yields an average of 188.3 nodes and 105.0 edges, nearly twice the scale observed in GAIA (99.2 nodes, 61.7 edges) and xbench-

Metric	Method	BrowseComp	BrowseComp-ZH	GAIA	xbench-DeepSearch
Avg. LLM Input Tokens	Flash-Searcher	2,866,808.4	1,785,539.7	2,235,279.5	1,957,544.3
	STRUCTMEM	2,696,703.3	862,501.7	424,029.4	781,391.3
Avg. LLM Output Tokens	Flash-Searcher	584,006.3	335,438.5	420,715.2	412,051.7
	STRUCTMEM	592,254.8	235,974.2	168,187.6	233,205.3
Avg. Retrieval Calls	Flash-Searcher	301.0	142.6	171.8	156.7
	STRUCTMEM	380.4	194.9	126.1	151.8

Table 2: Average execution cost comparison between Flash-Searcher and STRUCTMEM across four benchmarks.

DeepSearch (99.2 nodes, 57.9 edges). This demonstrates that STRUCTMEM adaptively expands in proportion to task complexity: harder queries trigger more tool invocations and denser logical connections, while simpler queries avoid unnecessary computational overhead.

Information compression. The bottom panel of Figure 3 shows that while raw observation nodes are numerous (e.g., 91.2 per task in BrowseComp), the number of integrated fact nodes remains small and stable (averaging only 2–3 per task). This highlights the effect of the semantic relational integrator: it consolidates observations, removes redundancy, and extracts a compact set of structured facts, yielding a concentrated knowledge representation that filters irrelevant information effectively.

5.3 RQ3: Efficiency analysis

To analyze the computational efficiency of STRUCTMEM, we compare its average execution cost with the strongest baseline Flash-Searcher across four deep-search benchmarks. We report three types of cost: LLM input tokens, LLM output tokens, and retrieval calls. Both methods are evaluated under the same setting using GPT-5 as the base model, and the results are shown in Table 2.

LLM input tokens. STRUCTMEM uses fewer LLM input tokens than Flash-Searcher across all four benchmarks. The reduction is particularly clear on BrowseComp-ZH (-51.7%), GAIA (-81.0%), and xbench-DeepSearch (-60.1%), suggesting that structured memory effectively compresses long interaction histories into compact evidence and reduces repeated context processing.

LLM output tokens. STRUCTMEM reduces LLM output tokens on most benchmarks, especially on GAIA (-60.0%) and xbench-DeepSearch (-43.4%). This indicates that structured memory supports evidence synthesis without introducing excessive generation overhead. On BrowseComp, its output-token cost is comparable to Flash-Searcher due to more active exploration, but it achieves higher accuracy, showing a better effectiveness–cost trade-off.

Retrieval calls. STRUCTMEM performs more retrieval on BrowseComp and BrowseComp-ZH, but fewer calls on GAIA and xbench-DeepSearch. This suggests that our method can still conduct active exploration when the task is difficult, while reducing redundant search when the accumulated structured memory is sufficient.

Overall, these results show that STRUCTMEM can achieve better performance while reducing computational cost. This demonstrates the advantage of our structured memory design.

5.4 RQ4: Component ablation

To further understand the contribution of each key component in STRUCTMEM, we conduct a component ablation study on the xbench-DeepSearch benchmark. We mainly examine two core modules: the epistemic gap analyzer and the semantic relational integrator.

Model Variant	xbench-DeepSearch
w/o Both	61.0
w/o Epistemic Gap Analyzer	64.0
w/o Semantic Relational Integrator	80.0
STRUCTMEM	**87.0**

Table 3: Component ablation of STRUCTMEM on the **xbench-DeepSearch** benchmark.

As shown in Table 3, removing both components reduces performance from 87.0 to 61.0, demonstrating the importance of combining structured fact integration with active reasoning guidance. Removing the epistemic gap analyzer causes a larger drop to 64.0, highlighting the role of explicit gap analysis in targeted exploration. Removing the semantic relational integrator also lowers performance to 80.0, showing that relational integration helps consolidate noisy observations into compact and consistent memory. Overall, the two modules are complementary and jointly contribute to the best performance of STRUCTMEM.

5.5 RQ5: Backbone model ablation

To disentangle the contribution of STRUCTMEM from the underlying LLM capability, we conduct a backbone ablation by instantiating the framework with models of varying strengths, while keeping

Model	MedXpertQA Text		
	Reasoning	Understanding	Average
Human Expert (Pre-Licensed)	41.7	45.4	42.6
GPT-4o	30.6	29.5	30.4
GPT-5-mini	45.9	43.8	44.9
GPT-5 (Full Set)	57.0	54.8	55.9
GPT-5 (Sampled Set)	56.0	55.0	55.5
STRUCTMEM	**63.0**	**68.0**	**65.5**

Table 4: Generalization performance on the **MedXpertQA** medical reasoning benchmark.

Model	Standard Inference	STRUCTMEM
Gemini-2.5-flash	8.0	35.0
GPT-4.1-mini	11.0	60.0
GPT-5-mini	17.0	78.0
GPT-5 (Default)	30.0	**87.0**

Table 5: Performance comparison between standard inference and STRUCTMEM with different base models on the **xbench-DeepSearch** benchmark.

all other components unchanged. Experiments are performed on the xbench-DeepSearch benchmark.

As shown in Table 5, we find that: (i) STRUCTMEM performs strongly across all backbone models, including lightweight LLMs, indicating that its effectiveness does not rely on large-scale parametric knowledge; (ii) performance improves steadily with model capability (from 35.0 to 87.0), showing that STRUCTMEM scales synergistically with stronger backbones; and (iii) the strong performance of smaller models (e.g., GPT-5-mini reaching 78.0) highlights that STRUCTMEM provides a robust external reasoning substrate that compensates for limited model capacity. This confirms that the gains stem from structured reasoning and memory organization, rather than solely from the backbone LLM. Overall, these results demonstrate that STRUCTMEM is a backbone-agnostic framework that consistently enhances reasoning performance, offering both strong effectiveness with small models and scalable gains with larger ones.

5.6 RQ6: Model generalization analysis

To assess the cross-domain generalization ability of STRUCTMEM, we evaluate the framework on the **MedXpertQA** benchmark (Zuo et al., 2025), which requires expert-level medical reasoning and deep understanding (Wang et al., 2025). We focus on the text-only version, consisting of two question types: Reasoning and Understanding. Due to computational constraints, we randomly sampled 100 instances from each subset. To confirm representativeness, we benchmark GPT-5 on this sample, obtaining 56.0% on Reasoning and 55.0% on Understanding, closely matching full-set results (57.0% and 54.8%). This validates that the subset provides a reliable basis for comparison.

Table 4 reports the results. STRUCTMEM achieves the highest overall average score (65.5%), surpassing both GPT-5 (55.9%) and the pre-licensed human expert baseline (42.6%). This demonstrates that the framework generalizes effectively beyond its original web-search domain, supporting reasoning over specialized, knowledge-intensive content. The largest improvement is observed on the Understanding subset (+13.2% over GPT-5), suggesting that tasks requiring synthesis of complex concepts benefit from STRUCTMEM's dynamic structured memory, which organizes scattered evidence into coherent, relational representations. Overall, these results confirm that dynamically structuring knowledge not only enhances performance in open-world environments but also translates into robust domain generalization for specialized reasoning tasks.

6 Conclusion

We have introduced STRUCTMEM, a novel agent framework for structured reasoning on complex, long-horizon tasks. Unlike conventional agents that accumulate unstructured interaction traces, STRUCTMEM maintains a compact, dynamically evolving memory by constructing knowledge graphs on-the-fly. The semantic relational integrator distills and organizes noisy observations into a concise set of structured facts, while the epistemic gap analyzer identifies unresolved knowledge gaps, generates targeted exploratory queries, and guides the agent's reasoning in real time.

Extensive experiments show that STRUCTMEM achieves state-of-the-art performance across deep-search benchmarks, improves execution efficiency, and generalizes to specialized domains without task-specific fine-tuning. Overall, STRUCTMEM demonstrates that structured memory can serve as a scalable and effective abstraction layer for inference-time reasoning.

Limitations

Despite its strong performance, StructMem introduces several limitations. (i) *Error propagation in structured abstraction.* The framework relies on the underlying LLM to extract and structure facts from noisy observations. When the backbone model produces inaccurate or incomplete abstractions, these errors can be explicitly encoded into the structured memory and propagate through subsequent reasoning steps. While structuring improves interpretability and efficiency, it also makes errors more persistent once incorporated into the memory. (ii) *Temporal information handling.* The current design does not explicitly model temporal validity in the structured memory graph. As a result, facts collected at different times may be treated similarly, even when some information is stale or outdated. This limitation can affect tasks that require up-to-date evidence or reasoning over time-sensitive knowledge.

Future work will explore more robust and efficient structured reasoning mechanisms. On the robustness side, we plan to incorporate uncertainty-aware fact representations (e.g., confidence-weighted or probabilistic edges) and develop verification strategies based on cross-source agreement or multi-path reasoning consistency, enabling the agent to detect and revise unreliable facts dynamically. On the temporal reasoning side, we plan to augment structured facts with timestamps and validity ranges, enabling the agent to identify, prioritize, and revise time-sensitive knowledge. More broadly, an interesting direction is to enable self-refining memory, where the agent continuously revises and reorganizes its structured knowledge over time, potentially improving both accuracy and long-horizon reasoning efficiency.

Ethical Considerations

In this paper, all the foundation models and tools used in our experiments are publicly accessible. All experimental data and evaluations are based on publicly available benchmarks, and no new personal information is exposed or collected in this process, ensuring that user privacy is strictly protected. Additionally, the StructMem framework we propose aims to enhance the effectiveness, transparency, and factual grounding of autonomous agents during complex reasoning tasks. By explicitly structurally resolving factual conflicts, our approach actively mitigates LLM hallucinations and does not encourage or induce the model to produce any harmful, biased, or misleading information. Therefore, We believe our work complies with ACL ethical guidelines.

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17682–17690.

Théophile Bieth, Yoed N Kenett, Marcela Ovando-Tellez, Alizée Lopez-Persem, Célia Lacaux, Marie Scuccimarra, Inès Maye, Jade Sénéchal, Delphine Oudiette, and Emmanuelle Volle. 2024. Changes in semantic memory structure support successful problem-solving and analogical transfer. *Communications Psychology*, 2(1):54.

ByteDance Doubao. 2025. Doubao. http://www.doubao.com/.

Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, and 1 others. 2025. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations. *arXiv preprint arXiv:2506.13651*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

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*.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. 2025. Memp: Exploring agent procedural memory. *arXiv preprint arXiv:2508.06433*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2023. A real-world webagent with plan-

ning, long context understanding, and program synthesis. *arXiv preprint arXiv:2307.12856*.

Inc. Jina. 2025. Jina reader. https://jina.ai/reader/.

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*.

Tomoyuki Kagaya, Thong Jing Yuan, Yuxuan Lou, Jayashree Karlekar, Sugiri Pranata, Akira Kinose, Koki Oguri, Felix Wick, and Yang You. 2024. Rap: Retrieval-augmented planning with contextual memory for multimodal llm agents. *arXiv preprint arXiv:2402.03610*.

Graham Klyne. 2004. Resource description framework (rdf): Concepts and abstract syntax. *http://www. w3. org/TR/rdf-concepts/*.

Bang Liu, Xinfeng Li, Jiayi Zhang, Jinlin Wang, Tanjin He, Sirui Hong, Hongzhang Liu, Shaokun Zhang, Kaitao Song, Kunlun Zhu, and 1 others. 2025. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*.

MiroMind AI Team. 2025. Miroflow: An open-source agentic framework for deep research. https://github.com/MiroMindAI/MiroFlow. GitHub repository.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, and 1 others. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

OpenAI. 2025. Deep research system card. https://cdn.openai.com/deep-research-system-card.pdf.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T Le, Samira Daruki, Xiangru Tang, and 1 others. 2025. Reasoningbank: Scaling agent self-evolving with reasoning memory. *arXiv preprint arXiv:2509.25140*.

Tianrui Qin, Qianben Chen, Sinuo Wang, He Xing, King Zhu, He Zhu, Dingfeng Shi, Xinxin Liu, Ge Zhang, Jiaheng Liu, and 1 others. 2025. Flash-searcher: Fast and effective web agents via dag-based parallel execution. *arXiv preprint arXiv:2509.25301*.

Serper, Inc. 2025. Serper api. https://serper.dev/.

Zhengliang Shi, Yiqun Chen, Haitao Li, Weiwei Sun, Shiyu Ni, Yougang Lyu, Run-Ze Fan, Bowen Jin, Yixuan Weng, Minjun Zhu, Qiujie Xie, Xinyu Guo, Qu Yang, Jiayi Wu, Jujia Zhao, Xiaqiang Tang, Xinbei Ma, Cunxiang Wang, Jiaxin Mao, and 7 others. 2025a. Deep research: A systematic survey. *Preprint*, arXiv:2512.02038.

Zhengliang Shi, Lingyong Yan, Dawei Yin, Suzan Verberne, Maarten de Rijke, and Zhaochun Ren. 2025b. Iterative self-incentivization empowers large language models as agentic searchers. *arXiv preprint arXiv:2505.20128*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, and 1 others. 2025. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. Scaling llm test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*.

Tongyi DeepResearch Team. 2025. Tongyi-deepresearch. GitHub repository.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 10014–10037.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Shansong Wang, Mingzhe Hu, Qiang Li, Mojtaba Safari, and Xiaofeng Yang. 2025. Capabilities of gpt-5 on multimodal medical reasoning. *arXiv preprint arXiv:2508.08224*.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia

Glaese. 2025a. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*.

Wenda Wei, Yu-An Liu, Ruqing Zhang, Jiafeng Guo, Lixin Su, Shuaiqiang Wang, Dawei Yin, Maarten de Rijke, and Xueqi Cheng. 2025b. Thinking forward and backward: Multi-objective reinforcement learning for retrieval-augmented reasoning. *arXiv preprint arXiv:2511.09109*.

xAI. 2025a. Grok 3 beta — the age of reasoning agents. https://x.ai/news/grok-3.

xAI. 2025b. Grok 4 — the most intelligent model in the world. https://x.ai/news/grok-4.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy S Liang, and Jure Leskovec. 2022. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems*, 35:37309–37323.

Weihao Zeng, Keqing He, Chuqiao Kuang, Xiaoguang Li, and Junxian He. 2026. Pushing test-time scaling limits of deep search with asymmetric verification. In *The Fourteenth International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023a. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2023b. Synapse: Trajectory-as-exemplar prompting with memory for computer control. *arXiv preprint arXiv:2306.07863*.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, and 1 others. 2025. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*.

Yihua Zhu, Qianying Liu, Akiko Aizawa, and Hidetoshi Shimodaira. 2025. Beyond chains: Bridging large language models and knowledge bases in complex question answering. *arXiv preprint arXiv:2505.14099*.

Yuxin Zuo, Shang Qu, Yifei Li, Zhangren Chen, Xuekai Zhu, Ermo Hua, Kaiyan Zhang, Ning Ding, and Bowen Zhou. 2025. Medxpertqa: Benchmarking expert-level medical reasoning and understanding. *arXiv preprint arXiv:2501.18362*.

A Appendix

A.1 Case study

We present a representative multi-hop reasoning trajectory from BrowseComp (Task ID: 28) in Table 6 to provide a qualitative analysis of StructMem.

Strengths. This case illustrates how StructMem transforms large-scale, noisy web interactions into concise and actionable structured knowledge. Starting from 124 raw observations collected via search and browsing, the framework distills only a small set of key fact triples through fact extraction and relational integration. This structured memory explicitly captures core entities and their relations, enabling the agent to perform stable multi-hop reasoning without being distracted by irrelevant context. Compared to trajectory-based methods over unstructured histories, this yields better information compression and reasoning focus.

Weaknesses. Noisy observations may still be abstracted into spurious fact triples, which can propagate and affect downstream reasoning. Thus, while structured memory improves robustness, it does not fully eliminate error accumulation, especially when early extraction steps are inaccurate. This suggests the need for more reliable fact validation and noise filtering mechanisms (see Section Limitations).

Overall, StructMem effectively condenses large-scale, noisy web observations into concise structured knowledge for accurate reasoning, while leaving robustness as a key area for improvement.

Component	Content detail
Complex task *(Multi-hop query)*	A TV show in the 1960s was produced by a TV station, channel 8, which, as of 2023, ran programming from the PBS network, in a city in the US that shares its name with a city in Southern Europe. The show's presenter interacted with cartoon characters, and the name of the show includes a food item. In the series, the presenter's first name was only two letters. The show had an episode whose title mentioned a trip to a specific destination. **What was this destination?**
Initial plan & goal *(Nodes: 3)*	**Initial plan:** Targeted PBS station verification. Use web search with queries like "WGTV channel 8 Athens PBS" to identify WGTV as a PBS member station located in Athens, GA (sharing a name with Athens, Greece)... **Goal 1:** Identify the qualifying PBS "channel 8" station and its city that shares a name with a city in Southern Europe... **Goal 2:** ...
Raw observations *(Nodes: 124) [Highly noisy]*	*The agent executed 124 search/crawl actions and accumulated massive unstructured text buffers:* • `Obs_2`: [WGTV CH 8 | Georgia Public Broadcasting] (https://www.gpb.org/...) • `Obs_4`: [Georgia Greats | Dean Rusk: At the Heartbeat of History - PBS] ... • `Obs_24`: [PDF] ED119715 - Produced by WGTV-TV, Athens GA; episode list; A Trip to the University; series title; host... • `Obs_37`: TV Guide Archive - PBS Schedule 1974; WGTV programming lineup; evening news; documentary slot; local broadcast schedule... *... (121 other lengthy webpage records, historical archives, and irrelevant TV guides omitted)*
Structured facts *(Nodes: 5) [Significant compression]*	StructMem*'s Semantic Relational Integrator distilled the 124 noisy web pages into exactly 5 high-value factual triples:* 1. ⟨Show episode, *title*, "A Trip to the University"⟩ 2. ⟨"A Trip to the University", *destination*, **"the University"**⟩ 3. ⟨Show episode 21, *title*, "Visit to the World of the Muffins, Part I"⟩ 4. ⟨Miss Jo and her cartoon drawings, *take children to*, the curious world of Muffin land⟩ 5. ⟨The show, *encourages*, 3 to 7-year-olds to use imagination⟩
Agent's prediction	**the University**
Standard answer	**University** *(Correct match)*

Table 6: Case study from the BrowseComp benchmark. StructMem successfully prevents context window overflow by distilling 124 unstructured and noisy webpage observations into just 5 explicit factual triples. These structured facts enable the agent to accurately deduce the standard answer to a highly complex multi-hop query. Due to space limitations, we only show several key sentences from the reasoning process.