
MEM1: Learning to Synergize Memory and Reasoning for Efficient Long-Horizon Agents

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 Modern language agents must operate over long-horizon, multi-turn interactions,
2 where they retrieve external information, adapt to observations, and answer in-
3 terdependent queries. Yet, most LLM systems rely on full-context prompting,
4 appending all past turns regardless of their relevance. This leads to unbounded
5 memory growth, increased computational costs, and degraded reasoning perfor-
6 mance on out-of-distribution input lengths. We introduce **MEM1**, an end-to-end
7 reinforcement learning framework that enables agents to operate with **constant**
8 **memory** across long multi-turn tasks. At each turn, MEM1 updates a **compact**
9 **shared internal state** that jointly supports memory consolidation and reasoning.
10 This state integrates prior memory with new observations from the environment
11 while strategically discarding irrelevant or redundant information. To support
12 training in more realistic and compositional settings, we propose a simple yet
13 effective and scalable approach to constructing multi-turn environments by com-
14 posing existing datasets into arbitrarily complex task sequences. Experiments
15 across three domains, including internal retrieval QA, open-domain web QA, and
16 multi-turn web shopping, show that MEM1-7B improves performance by $3.5\times$
17 while reducing memory usage by $3.7\times$ compared to Qwen2.5-14B-Instruct on a
18 **16-objective** multi-hop QA task, and **generalizes beyond the training horizon**.
19 Our results demonstrate the promise of reasoning-driven memory consolidation
20 as a scalable alternative to existing solutions for training long-horizon interactive
21 agents, where both efficiency and performance are optimized.

22 1 Introduction

23 Large language models (LLMs) have shown remarkable performance in single-turn tasks such
24 as question answering, summarization, and code generation [7, 48, 3]. However, emerging real-
25 world applications increasingly operate over multiple turns—searching documents, interacting with
26 environments [67], and making decisions based on evolving external information [50]. Examples
27 include research agents such as OpenAI and Gemini Deep Research [34, 16] that automate complex
28 tasks by iteratively gathering information, and web-navigation agents such as OpenManus [47] and
29 BrowerUse [31], which must complete goals across dozens of interactive turns.

30 Unlike traditional tasks where the input is static or self-contained, long-horizon settings often
31 involve answering a sequence of related questions, requiring the agent to continuously retrieve new
32 information, revise beliefs, and adapt to evolving contexts over time. For instance, consider a research
33 assistant tasked with “What’s the evidence for X?”. Subsequent queries like “Who published it?”
34 require further information retrieval, while “Is the source credible?” calls for self-reflection and
35 assessment. Each query builds on the previously collected and accumulated information. Similarly, a
36 shopping assistant may be first asked “Which product is cheapest?”, then “What are its reviews?”,

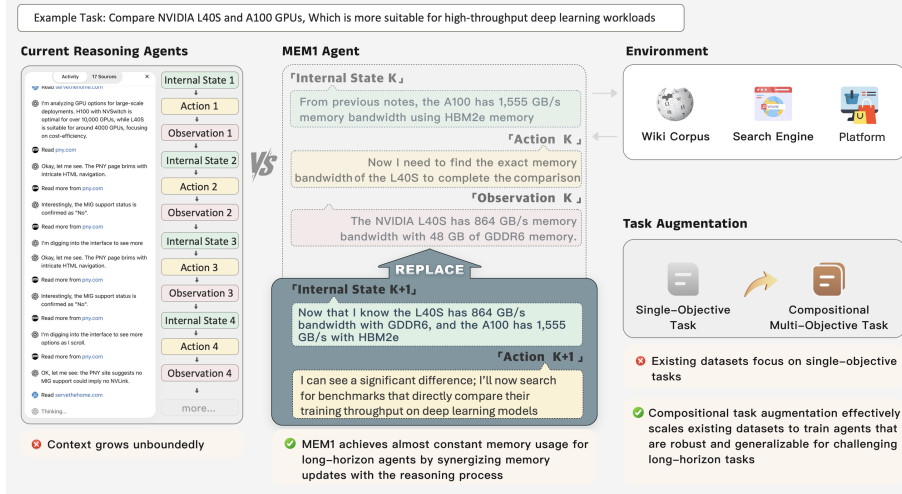


Figure 1: A comparison between MEM1 and current reasoning agents in memory management.

and “Is it compatible with my device?”. These interactions span multiple turns, featuring evolving contexts and compound reasoning.

In systems designed for long-horizon settings, a common approach is to append all past observations, actions, and thoughts to the context at every turn [52, 58]. This forces the model to operate with an unboundedly growing context, which introduces three key challenges. **(1) Growing inference cost and memory usage.** Transformer-based LLMs typically incur $O(N^2)$ compute cost (or $O(N)$ with Key-Value caching) and $O(N)$ memory usage as the context length N increases [49]. Consequently, deploying these models requires reserving large GPU memory on modern inference frameworks to accommodate the growing context [25, 65], often leading to significant wastage of computing resources. **(2) Generalization limits beyond the training horizon.** Ongoing conversations with context length exceeding that in the training data become out-of-distribution for the model. The model struggles to manage and reason over such unfamiliar long-horizon inputs [60]. **(3) Overloaded and inefficient context.** The accumulation of irrelevant or redundant content dilutes the model’s attention. This reduces its ability to reason effectively, even when relevant information is technically still present within the prompt [2, 27, 53].

Recent progress in long-context modeling largely targets static inputs (e.g., long documents) and does not address multi-turn interaction with external environments [6, 17]. Some other approaches introduce external memory modules (e.g., summarizers or retrievers) [60, 26, 11, 54], but these are typically trained separately and cannot be optimized end-to-end with the agent’s policy. This also introduces additional engineering overhead, as engineers must manage and integrate two separate models. Meanwhile, existing works on tool-using agent systems trained with reinforcement learning leave memory management unsolved, letting the prompt length grow unboundedly [22, 66]. A natural question is raised: *Can a language model learn to consolidate its memory as part of its reasoning process so that it retains only what is essential for solving the task?*

Motivated by this question, we present MEM1: **Memory-Efficient Mechanism** via learning 1-step integrated reasoning and consolidation—a method for training LLM agents that maintain constant memory usage across arbitrarily long horizons. As illustrated in Fig. 1, at each turn, the model updates a consolidated state composed of prior memory and newly obtained information. This consolidated state becomes the agent’s only retained memory, allowing all external tool outputs to be discarded after use, which prevents prompt expansion altogether. A key insight of our method is that inference-time reasoning [52, 13, 30, 62] serves a dual function: it not only provides deeper insight into the current query but also acts as a form of “working memory” [4], extracting key components from gathered information to build an evolving understanding. By unifying reasoning and memory consolidation, MEM1 offers an elegant solution where the agent learns to reason and remember within a shared representational space without requiring additional modules or architectural changes.

We train this behavior end-to-end with reinforcement learning (RL) [45, 68], optimizing for task success via verifiable rewards [40]. Although we do not explicitly optimize for memory efficiency

74 through reward signals, the agent learns to manage memory as part of its policy, resulting in near-
75 constant memory usage across long horizons. To evaluate our method comprehensively, we employ
76 diverse multi-turn environments, including internal retrieval-augmented QA [24, 56], open-domain
77 web QA [66], and complex multi-turn shopping scenarios in WebShop [57]. Across tests, our method
78 consistently rivals the performance of leading baselines while delivering efficiency gains up to $3.5\times$.

79 Despite the effectiveness of current training and evaluation environments, they predominantly focus
80 on single-objective tasks [24, 56, 35], limiting their ability to fully prepare agents for realistic, long-
81 horizon scenarios that inherently involve multiple sequential objectives. To address this limitation, we
82 introduce a scalable task augmentation approach, transforming existing single-objective QA datasets
83 into complex multi-objective tasks through compositions of N multi-hop questions. This approach
84 enables us to repurpose standard benchmarks in our community to more effectively train and evaluate
85 agents on long-horizon reasoning, an increasingly important capability in real-world applications. In
86 our experiments, agents trained on 2-objective compositions generalize effectively to tasks involving
87 up to 16-objective compositions. Notably, at the 16-objective level, our MEM1 achieves superior
88 accuracy compared to all baseline methods, along with $1.27\times$ lower peak memory usage and $1.78\times$
89 faster inference compared to the respective best uncollapsed baseline.

90 2 Related Work

91 **LLM agent with multi-turn environment.** LLM-based agents have evolved from handling single-
92 turn queries to serving as autonomous agents capable of multi-turn interactions such as web navi-
93 gation [57, 67] and complex research [66]. To enable such capabilities, Yao et al. [58] introduced
94 the ReAct (*i.e.*, Reason + Act) framework, which enhances LLMs’ ability to interact with external
95 environments by interleaving reasoning and action. Building on this prompting paradigm, subsequent
96 works have explored ways to improve agent performance through natural language feedback, enabling
97 iterative refinement [42, 29]. Recently, inference-time scaling has emerged as a promising direction
98 for enabling complex reasoning, with prior research incorporating evaluator (*e.g.*, verifier, reward
99 model) [43, 28] or world models [8]. In addition, there are two major lines of training approaches:
100 (1) behavior cloning (BC), which involves imitating expert trajectories to guide agent behavior by
101 supervised fine-tuning (SFT) [59, 14, 10], and (2) reinforcement learning (RL), which optimizes
102 agent policies by incentivizing desirable outcomes through rewards [44, 5, 36]. These methods aim to
103 align the agent’s behavior with task objectives, enabling more robust and generalizable performance.

104 **Memory management for LLM agents.** A widely adopted approach to memory management
105 in LLM-based agent systems involves appending all prior information—such as observations, in-
106 termediate thoughts, and actions—into the prompt at each interaction turn [58]. While this method
107 is straightforward and effective when the number of interactions required is small, it results in
108 unbounded context growth, leading to linearly scaled inference memory. Moreover, long contexts
109 often contain irrelevant or redundant information, which impairs the model’s reasoning capabili-
110 ties [2, 27, 53]. To mitigate these issues, recent studies have proposed external memory frameworks,
111 including retrieval-augmented generation and summarization modules [60, 26, 11, 54]. However,
112 these methods are typically trained or used independently of the agent’s policy, creating a disconnect
113 between memory and the reasoning process. In addition, managing and integrating such modules
114 often incurs extra computational overhead and system complexity. Despite these advancements, many
115 RL approaches for training LLM agents still rely on accumulating the full interaction history as
116 memory [22, 66, 36], leaving memory management during training an underexplored area. In this
117 work, we seek to bridge this gap by tightly integrating memory with the agent’s reasoning process,
118 thereby enabling more efficient and context-aware decision-making.

119 3 MEM1

120 Complex reasoning tasks often require an iterative process of information gathering and synthesis,
121 as seen in applications such as “deep research” [34, 21] and web-based agents [32, 18]. Recent
122 advances in agent design involve interaction loops that interleave chain-of-thought reasoning [52, 13],
123 environment interaction, and real-world feedback collection. To explicitly capture these core elements,
124 we annotate each component using XML-style tags: `<IS>` for internal state (reasoning), `<query>`
125 for environment queries, `<answer>` for the agent’s responses, and `<info>` for external observations

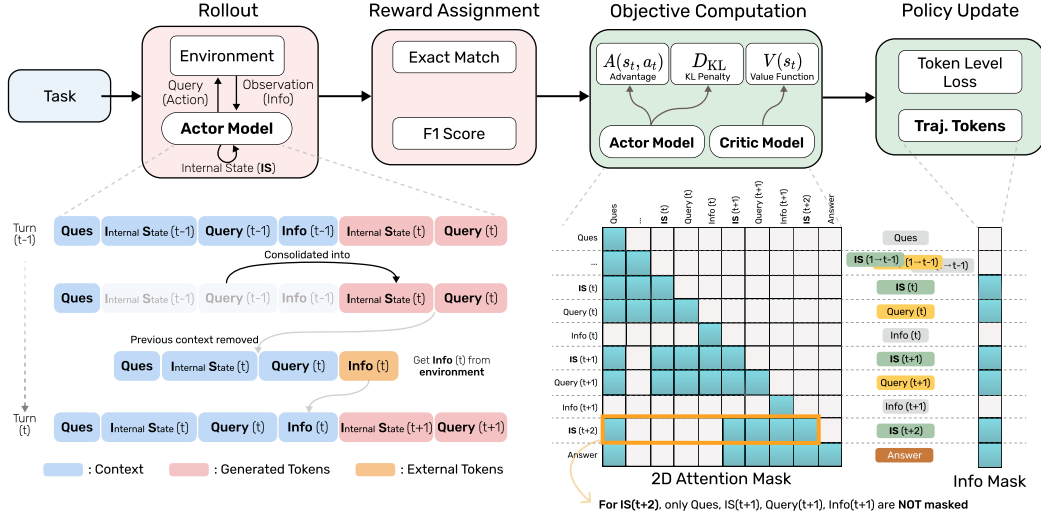


Figure 2: (Top): the RL pipeline used to train MEM1. (Bottom left): The evolution of context in MEM1—old `<IS>`, `<query>`, `<info>` are cleared as new states enter the context. The mechanism is used in the rollout. (Bottom right): the 2D attention mask used during the objective computation stage. The mask is applied to model forwarding when computing the log probabilities of the generated tokens; the information mask is then applied to the tokens in the full trajectory, masking out those not generated by the model itself during the policy update stage.

126 or tool outputs. MEM1 adopts a **learned** approach to iterative state updating and consolidation,
 127 ensuring that only the most recent set of `<IS>`, `<query>`, `<answer>`, and `<info>` elements is retained
 128 in the prompt at any given time. This design maintains a bounded and semantically relevant context,
 129 promoting efficient and coherent multi-step reasoning.

130 3.1 Memory as Part of Reasoning

131 To achieve a constant memory, MEM1 is particularly trained to iteratively refine its understanding
 132 by processing new information in conjunction with a consolidation of its prior state. At each turn t ,
 133 the agent produces a new `<ISt element, which summarizes past information and reasons about
 134 subsequent actions. Following this, the agent generates an inquiry—either a <queryt to interact
 135 with the environment, or an <answert if a direct response is warranted. If the agent issues
 136 a <queryt, the corresponding feedback from the environment is appended as <infot. At
 137 the next turn, $t + 1$, the agent consolidates the tuple $(\text{IS}_t, \text{query}_t, \text{info}_t)$ into a new
 138 <ISt+1, which serves as the basis for further interactions. After each turn, all tags from the
 139 previous turn t are pruned from the context, effectively compressing memory and preventing prompt
 140 bloat. Fig. 2 (bottom left) illustrates the evolution of the model’s context over time. At any given
 141 turn, the agent retains at most two <IS> elements, two <query> elements, and one <info> element,
 142 ensuring bounded and efficient memory usage. The detailed rollout algorithm is in Alg. 1 of App. A.5.`

143 RL offers a powerful mechanism for shaping agent behavior through reward signals [46]. In MEM1,
 144 we leverage this framework to incentivize effective state consolidation by designing environments
 145 in which the agent is rewarded only when it strategically retains and integrates useful information.
 146 Specifically, we construct tasks that require numerous interactions with the environment to arrive
 147 at a correct answer (see Sec. 3.3). Success depends on the agent’s ability to rely on information
 148 collected along the inference path. At each turn, we prune the agent’s context to retain only the
 149 most recent `<IS>`, forcing the agent to perform memory consolidation as part of its reasoning
 150 process. Without access to full historical context, the agent must learn to preserve and update relevant
 151 knowledge internally in order to reap the reward. This learning procedure mirrors how humans
 152 cultivate memorization skills through structured tasks such as Sudoku or crosswords [1], where
 153 success hinges on selectively attending to key information and building upon it. Over time, such
 154 tasks help individuals develop cognitive strategies that jointly support efficient memorization and
 155 reasoning, similar to our RL method for training MEM1.

3.2 Masked Trajectory for Policy Optimization

In the previous section, we detailed the rollout process of our RL pipeline. However, unlike conventional multi-turn agents that preserve a static context during generation, MEM1 introduces a unique challenge: its mechanism continuously updates the context at each turn by consolidating prior memory and pruning irrelevant tokens. This dynamic context update disrupts the continuity of the token generation trajectory, complicating the estimation of token-wise advantages in policy optimization algorithms such as PPO and Reinforce++ [38, 20], where trajectories are typically assumed to be linear.

To address this, we introduce a masked trajectory approach that reconstructs a logically coherent full trajectory by stitching together multiple interaction turns with evolving contexts. This unified trajectory mimics a standard multi-turn rollout and comprises a sequence of tuples $\tau_t = (\text{IS}_t, \text{query}_t, \text{info}_t)$ for $t \in [1, T - 1]$, where T denotes the total number of interaction turns. The T -th turn outputs the final answer $\tau_T = (\text{IS}_T, \text{answer}_T)$. The full trajectory encodes all information needed for accurate policy learning while respecting MEM1’s memory consolidation at each turn. Fig. 2 (bottom left) demonstrates the evolution of the agent’s context.

To ensure that policy gradients are correctly computed under this consolidated memory regime, we apply a **two-dimensional attention mask** [37] across the full trajectory. This mask restricts each token’s attention to only the tokens retained in memory at the time that token was generated. Specifically, for a token position k , we mask out all prior tokens that are not part of the consolidated memory corresponding to the context at that turn. This masking strategy enables accurate computation of the policy objective: letting s_k denote the masked input state and a_k the predicted token, the log-probability ratio $\rho_k(\theta) = \frac{\pi_\theta(a_k|s_k)}{\pi_{\theta_{\text{old}}}(a_k|s_k)}$ remains valid and tractable, in turn ensuring that the advantage, KL penalty, and value estimation are correct. A visualization is in Fig. 2 (bottom right). Furthermore, following [22], we incorporate a one-dimensional attention mask over retrieved external information during policy updates for both the actor and critic networks. This ensures that gradient updates are localized to only tokens generated by the agent. Fig. 2 (bottom right) shows the masking mechanism that enables stable and accurate policy optimization under MEM1’s memory-constrained execution.

3.3 Multi-Objective Task Design

Although our proposed method is designed to address the critical challenges of agentic multi-turn interaction with the external world, there are limited publicly available datasets that support training for such long-horizon interactive processes. Existing benchmarks, such as HotpotQA [56], Bamboogle [35], and 2wiki [19], are often cited as multi-hop benchmarks, yet they typically involve only two information-seeking steps. Moreover, these datasets are not explicitly structured to support long-horizon interactions that necessitate the agent to manage the memory state.

To bridge this gap, we introduce a novel task—multi-objective question & answering (QA)—that extends the number of reasoning steps required to solve a problem. Building on existing multi-turn datasets such as HotpotQA and Natural Question [56, 24], we interleave multiple questions from the original QA corpus and construct a single composite query that requires answering all constituent sub-questions, shown in Prompt 1 of App. A.3. This formulation compels the agent to perform multiple search queries, each targeting a distinct sub-question or sub-objective, and then integrate the retrieved answers to form a comprehensive final response. Compared to the original tasks, our synthesized multi-objective, multi-hop setting significantly increases the number of search and reasoning turns required, thereby promoting more complex, memory-intensive interactions.

4 Experiments & Results

We empirically demonstrate the effectiveness of our approach in training the MEM1 agent to perform multi-turn tasks while preserving a fixed-size memory state. We evaluate MEM1 against several baselines using a comprehensive set of metrics categorized into *accuracy* (e.g., Exact Match, F1 score, Environment Reward) and *efficiency* (e.g., Peak Token Usage, Dependency Length, Inference Time). All MEM1 variants are fine-tuned from the Qwen2.5-7B Base model [55]. We use PPO [38] as the RL algorithm as it computes token-level advantages, bringing stability to the training process. While we also experimented with instruction-tuned and supervised fine-tuned models using curated

high-quality trajectories, reinforcement learning from the base model consistently yielded the best performance and generalization.

Our experiments are conducted in two standard environments, each reflecting real-world scenarios that require multi-turn agent interactions. The first environment is question answering with retrieval-augmented generation (RAG) [24, 56], where the agent must answer queries by retrieving relevant information from an external knowledge store (either a database or an online search engine). We trained on RAG with local database (*i.e.*, Wikipedia Corpus) and evaluated on tasks with open web browsing. For QA, following Sec. 3.3, we construct multi-objective tasks and tested the model performance on tasks with more questions than seen in the training. The second environment is WebShop navigation [57], where the agent assists users in online shopping by browsing a website and selecting items based on natural language descriptions. This task requires the agent to iteratively read page content and make navigation decisions, following protocols similar to those in WebGPT [32].

4.1 Implementation Details

Datasets and evaluation metrics. We train two versions of MEM1 agent for both long-horizon QA and web navigation. For long-horizon QA, we augment the multi-hop QA dataset from [22] that mixes data from both HotpotQA [56] and Natural Question [24] to form a 2-objective composite task. For web agent, we use the WebShop environment [57], which also produces a reward during training [61]. For all datasets, the train-test split follows the original papers. During RL training, we employ the exact match (EM) metric for QA tasks (details in App. A.4) and the environment reward for Webshop [57, 61]. To evaluate the effectiveness of various approaches, we measure the EM and F1 score for QA tasks and final reward for the WebShop environment [57, 61]. To evaluate the efficiency, we consider the peak token usage, average dependency, and average inference time. The test datasets are obtained from the original papers which consist of out-of-distribution data. The former two metrics measure the memory efficiency, while the latter measures the time efficiency. The detailed definitions of the metrics are in App. A.4. The prompt and format can be found in App. A.3.

Baselines. To evaluate the accuracy and efficiency of MEM1, we compare it against a diverse set of baselines designed to either enhance task performance or manage context effectively. For the QA environment, we benchmark accuracy against Search-R1 [22], Deep Researcher [66], and a larger-scale model, Qwen2.5-14B-Instruct [55]. For the WebShop environment, we compare against Agent-FLAN [9], Agent-R [61], and AgentLM [63]. To assess efficiency, we consider two context compression baselines using models of the same parameter size as MEM1. First, we apply MEM1’s agentic *truncation* prompt template and rollout to a standard instruct model, isolating the benefits of prompt and rollout design alone. Second, we evaluate A-MEM [54], which augments an Instruct model with a vector database for memory retrieval, capturing the effect of external memory modules in agentic systems. We additionally train a supervised fine-tuned (SFT) model using trajectories curated from GPT-4o [33] based on MEM1’s rollout and compare it with the RL-trained agent.

Meta info injection. In our agentic pipeline, the agent’s context is programmatically truncated at each turn—immediately after it generates a search query or an answer—following the procedure outlined in Sec. 3. As past context is truncated, the agent may have difficulty determining when to terminate. To address this, we prepend a hint [HINT: YOU HAVE {turns_left} TURNS LEFT] at the beginning of each <info> tag to remind the agent of its remaining turns budget. For all experiments, we set the maximally allowed turns to 6 for 1-objective to 4-objective tasks and 20 for more difficult tasks to avoid excessively long trajectories.

4.2 MEM1 on Multi-Objective Multi-Hop Tasks

One key advantage of MEM1 agents lies in their efficient management of long-horizon interactions with the environment. To demonstrate this, we train our MEM1 agent with a 2-objective augmentation of the QA dataset, and subsequently test it against other models, using held-out multi-objective test datasets similarly augmented from the original test datasets. As elaborated in Sec. 3.3, these multi-objective tasks require a significantly larger number of turns of environment interactions to complete, hence serving as better benchmarks for memory management. As shown in Tab. 1, when evaluated on 2-objective datasets, MEM1 achieves comparable effectiveness (in terms of EM and F1 scores) to other 7B counterparts, while incurring significantly lower peak token usage and faster inference time.

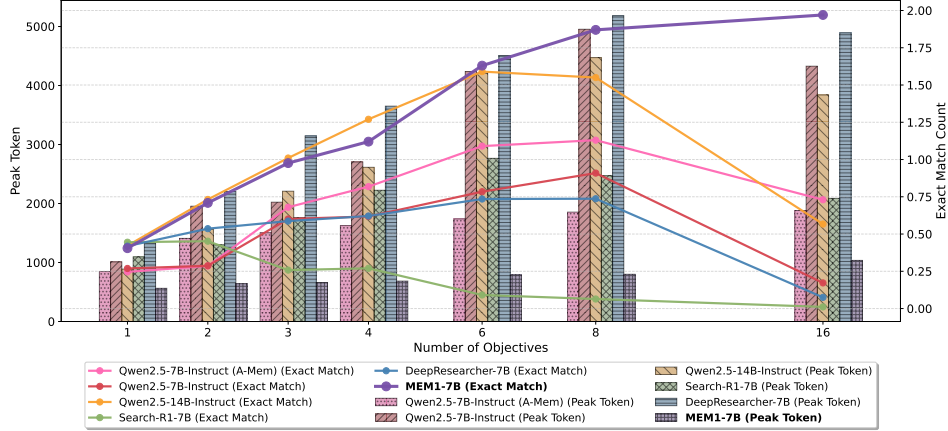


Figure 3: Performance and efficiency scaling of MEM1 (trained on 2-objective QA) with the number of objectives in multi-objective tasks. MEM1 outperforms the other models and baselines while having an almost constant scaling in memory usage. Note that at 16-objective, the context of baseline models does not increase anymore since their model performance has degraded (some collapsed).

Table 1: Comparison of models on multi-objective multi-hop QA tasks. Arrows indicate the desired directions. Numbers in red indicate collapsed model behavior. (truncate) means using MEM1’s prompt and rollout pipeline. (A-MEM) means using MEM1’s prompt and rollout pipeline with A-Mem’s external memory module [54]. MEM1-QA means MEM1 trained on 2-objective QA task.

Model	2-Objective				8-Objective				16-Objective			
	EM \uparrow	F1 \uparrow	Peak ($\times 10^2$) \downarrow	Time (s) \downarrow	EM \uparrow	F1 \uparrow	Peak ($\times 10^2$) \downarrow	Time (s) \downarrow	EM \uparrow	F1 \uparrow	Peak ($\times 10^2$) \downarrow	Time (s) \downarrow
Qwen2.5-14B-Inst	0.732	0.902	15.6 \pm 0.19	5.49 \pm 0.16	1.55	1.87	44.7 \pm 0.37	16.2 \pm 0.27	0.567	0.703	38.4 \pm 0.71	29.7 \pm 0.75
Qwen2.5-7B-Inst	0.268	0.366	19.6 \pm 0.33	4.60 \pm 0.08	0.87	1.10	49.5 \pm 0.40	13.9 \pm 0.18	0.165	0.213	43.3 \pm 0.62	15.5 \pm 0.23
Qwen2.5-7B-Inst (A-MEM)	0.286	0.371	14.1 \pm 0.10	24.6 \pm 0.51	1.13	1.43	18.6 \pm 0.10	53.7 \pm 1.26	0.730	0.961	18.8 \pm 0.14	91.2 \pm 2.44
Qwen2.5-7B-Inst (truncate)	0.262	0.336	8.28 \pm 0.06	5.89 \pm 0.16	0.97	1.23	11.8 \pm 0.10	11.9 \pm 0.20	0.396	0.497	13.3 \pm 0.16	22.1 \pm 0.60
Search-R1	0.452	0.531	13.0 \pm 0.08	4.09 \pm 0.23	0.064	0.08	24.7 \pm 0.19	4.25\pm0.16	0.009	0.011	20.9\pm0.03	4.75\pm0.18
DeepResearcher	0.536	0.650	22.0 \pm 0.43	4.01\pm0.07	0.73	0.90	51.8 \pm 0.35	11.3 \pm 0.14	0.071	0.106	48.9\pm0.66	15.8\pm0.19
MEM1-QA	0.709	0.838	6.40\pm0.02	6.49 \pm 0.07	1.87	2.31	8.01\pm0.06	8.68\pm0.12	1.97	2.39	10.4\pm0.09	8.70\pm0.12

The advantage of MEM1 becomes even more evident in tasks requiring longer-horizon interactive processes. To highlight such scalability of MEM1, we further compare the models on 3, 4, 6, 8, and 16-objective tasks in Fig. 3 and Tab. 1. Fig. 3 illustrates the scaling trends of task performance (measured by EM count) and memory efficiency (measured by Peak Token Usage) for MEM1 relative to other models and memory management baselines. As the number of objectives increases, the Peak Token Usage of all other methods and models scales nearly linearly. In contrast, MEM1 maintains an almost constant peak token count with only a slight increase, as also shown in Tab. 1.

Notably, while MEM1 initially underperforms Qwen2.5-14B-Instruct, its performance gradually catches up as the number of objectives increases, eventually surpassing the 14B model, which has double the parameter count. MEM1 also demonstrates remarkable efficiency. In the 16-objective task, it requires only 27.1% of the peak tokens and 29.3% of the total inference time compared to Qwen2.5-14B-Instruct. This efficiency translates to significantly reduced GPU memory requirements and overall computing resource demands.

4.3 MEM1 on Single-Objective Multi-Hop Tasks

While MEM1 is designed to train agents for very long-horizon tasks, our training method also delivers improved capability with existing multi-hop tasks while achieving much greater efficiency at the same time. Note that single-objective tasks also require multiple turns of interaction to produce the desired output.

Long-horizon web navigation in WebShop. Beyond QA tasks, we further evaluate the effectiveness of MEM1 in managing long-horizon interactions in the form of web navigation. We show the experimental results in Tab. 2. Trained in the WebShop environment (see App. A.6), MEM1 outperforms other agent training baselines, including Agent-Flan, Agent-R and AgentLM when

Table 2: The experimental results for Webshop. For a fair comparison, we do not report GPT’s inference time. For Agent-R, scores are taken from the original paper as the model is closed source. MEM1-WebShop means MEM1 trained on WebShop environment.

Model	Avg Final Reward \uparrow	Peak Token ($\times 10^3$) \downarrow	Dependency ($\times 10^6$) \downarrow	Inference Time Per Traj (s) \downarrow
GPT-4o	25.48	5.30 ± 1.23	3.99 ± 1.16	N/A
GPT-4o (truncate)	13.82	0.99 ± 0.99	0.81 ± 0.23	N/A
GPT-4o (A-MEM)	24.50	1.84 ± 0.06	0.31 ± 0.11	N/A
Qwen2.5-7B-Instruct	18.42	5.64 ± 1.34	3.38 ± 0.89	12.31 ± 1.82
Qwen2.5-14B-Instruct	12.34	5.44 ± 0.92	3.30 ± 0.61	18.17 ± 2.32
Agent-FLAN-7B	40.35	3.37 ± 1.12	2.18 ± 1.62	9.95 ± 6.19
Agent-R-8B	63.91	N/A	N/A	N/A
AgentLM-7B	63.60	2.24 ± 0.40	0.28 ± 0.07	3.91 ± 1.07
AgentLM-13B	70.80	2.36 ± 0.46	0.30 ± 0.08	5.23 ± 1.59
MEM1-WebShop	70.87	0.81 ± 0.10	0.15 ± 0.16	2.61 ± 0.48

Table 3: Performance comparison across environments for single-objective tasks. Arrows indicate desired direction. (SFT) means training with SFT and applying MEM1’s prompt and rollout.

Environment	System	EM \uparrow	F1 \uparrow	Peak Token ($\times 10^2$) \downarrow	Dependency ($\times 10^5$) \downarrow	Inference Time \downarrow
Wiki RAG	Qwen2.5-7B-Inst (truncate)	0.287	0.382	6.28 ± 0.05	1.65 ± 0.04	2.26 ± 0.04
	Qwen2.5-7B-Inst (A-MEM)	0.246	0.373	8.47 ± 0.12	0.92 ± 0.03	11.2 ± 0.40
	Qwen2.5-7B-Inst	0.269	0.390	9.32 ± 0.19	1.17 ± 0.04	2.31 ± 0.04
	Qwen2.5-14B-Inst	0.422	0.534	8.89 ± 0.21	2.22 ± 0.10	6.73 ± 0.24
	Search-R1	0.445	0.516	11.0 ± 0.25	1.50 ± 0.05	2.23 ± 0.14
	DeepResearcher	0.419	0.503	13.3 ± 0.34	7.04 ± 0.33	3.86 ± 0.09
	MEM1-QA (SFT)	0.302	0.358	6.54 ± 0.05	3.30 ± 0.13	4.84 ± 0.21
Online Web-QA	MEM1-QA	0.405	0.471	5.63 ± 0.03	0.76 ± 0.02	3.79 ± 0.07
	Qwen2.5-7B-Inst	0.334	0.451	8.37 ± 0.18	1.39 ± 0.06	2.20 ± 0.04
	DeepResearcher	0.372	0.492	10.27 ± 0.19	2.86 ± 0.14	2.87 ± 0.06
	MEM1-QA	0.397	0.485	5.79 ± 0.06	0.44 ± 0.02	1.84 ± 0.03

utilizing models of similar size. Furthermore, MEM1 achieves remarkable efficiency improvements compared to the best baseline method, AgentLM, featuring a $2.8\times$ improvement in Peak Token Usage, a $1.9\times$ improvement in Dependency, and a $1.5\times$ improvement in Inference Time. MEM1 even surpasses AgentLM-13B, a model with twice the parameter count of our trained model. Additionally, our results indicate that using MEM1 is significantly better than OpenAI’s GPT-4o on the WebShop tasks, even when the truncation prompt templates or A-MEM techniques are applied to GPT-4o.

Single-objective QA in Wikipedia. Tab. 3 presents the accuracy and efficiency metrics for evaluations on single-objective QA tasks on Wikipedia [22], where the agent can make retrieval requests from the Wikipedia datastore via RAG. Overall, MEM1 demonstrates superior efficiency across all three evaluated efficiency metrics, while simultaneously achieving the highest EM score and an F1 score comparable to that of Qwen2.5-14B-Instruct. This improvement in efficiency is attributed to the MEM1 agent’s ability to consolidate memory from previous interactions into a compact internal state, which reduces the number of tokens used in the context. We also observe that SFT significantly underperforms RL, highlighting the necessity for RL-based training.

Zero-shot transfer to Online Web-QA. To validate the transferability and generalizability of the trained MEM1 agent, we perform a zero-shot transfer to an Online Web-QA environment, which is unseen by the agent. In this environment, agents conduct web searches through an API service that returns results including titles, snippets and URLs. As shown in Tab. 3, MEM1 consistently exhibited improved efficiency alongside comparable effectiveness in this unseen setting via zero-shot transfer.

4.4 Analysis on Emergent Agent Behaviors

Through analyzing MEM1’s multi-turn interaction traces trained on 2-objective QA, we observe a range of emergent behaviors that are critical for handling long-horizon, multi-objective tasks, demonstrating capabilities well beyond simple retrieval. First, MEM1 learns to **manage multiple questions concurrently** by maintaining a structured internal state. As shown in Fig. 4(a), when faced with two multi-hop questions, the agent stores and updates memory for each question separately,

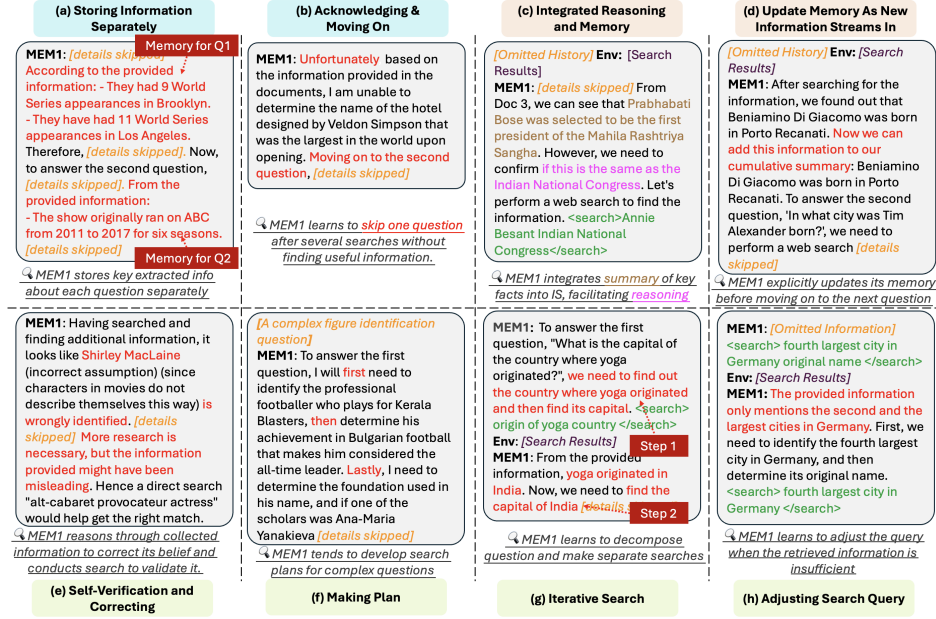


Figure 4: Snippets of internal states and actions showing MEM1’s Emergent Behaviors in 2-objective QA tasks. Light Blue denotes behaviors related to multi-objective tasks. Beige denotes behaviors related to memory in internal state. Pastel Green denotes behaviors related to general search strategies.

guiding subsequent searches based on the identified information gaps. In (b), MEM1 exhibits the ability to shift focus when progress on one question stalls, recognizing difficulty and prioritizing the more tractable objective. Meanwhile, MEM1 learns to **interleave reasoning and memory** in its internal state **<IS>**, weaving important information into its decision-making process to support both information retention and action selection. In Fig. 4 (c), MEM1 explicitly extracts important information from previous search results and leverages them to formulate the next query that best addresses the current information gap. In addition, (d) shows that when new, relevant information is retrieved, MEM1 explicitly reasons about its significance and selectively updates its memory. We believe that learning these interleaved behavior is key to achieving efficiency gain in memory without degrading performance. Beyond behaviors unique to our multi-objective setup and memory architecture, MEM1 also exhibits several **general-purpose search strategies**. In (e), the agent performs self-verification, correcting an earlier misconception and issuing a new query for confirmation. In (f), it decomposes complex queries into manageable subgoals before initiating search. In (g), for questions requiring multi-turn information gathering, MEM1 extracts key information from search results and uses it to inform the next search. In (h), when overly specific queries fail, MEM1 re-scopes its query to improve retrieval. Notably, many of these behaviors, including verification, making plan, and iterative search, are also reported in recent studies on deep research agents [22, 66].

5 Conclusion, Limitation, and Future Work

We introduced MEM1, a reinforcement learning framework that enables language agents to perform long-horizon reasoning with consolidated memory. By integrating inference-time reasoning and memory consolidation into a unified internal state, MEM1 addresses the scalability challenges of prompt growth and achieves competitive performance across QA and web navigation benchmarks, with substantially reduced memory usage and inference latency. Despite these advantages, MEM1 assumes access to environments with well-defined and verifiable rewards. While this assumption holds in domains such as QA, math, and web navigation, many open-ended tasks present ambiguous or noisy reward structures. Fully realizing the potential of MEM1 therefore requires advances in modeling such tasks and designing suitable reward mechanisms—challenges that lie beyond the scope of this work. A promising future direction is to explore methods for training MEM1 agents in these open-ended settings where reward signals are sparse, delayed, or implicit.

References

- [1] A Cognitive Connection. Surprising exercises that will sharpen your short-term memory, January 2024. URL <https://acognitiveconnection.com/surprising-exercises-that-will-sharpen-your-short-term-memory>. Accessed: 2025-05-10.
- [2] Chenxin An, Jun Zhang, Ming Zhong, Lei Li, Shansan Gong, Yao Luo, Jingjing Xu, and Lingpeng Kong. Why does the effective context length of llms fall short? In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.
- [3] Anthropic. The claude 3 model family: Opus, sonnet, haiku. <https://www.anthropic.com/news/claude-3-family>, 2024.
- [4] Alan D. Baddeley and Graham J. Hitch. Working memory. In Gordon H. Bower (ed.), *Psychology of learning and motivation*, volume 8, pp. 47–89. Academic Press, 1974.
- [5] Hao Bai, Yifei Zhou, Jiayi Pan, Mert Cemri, Alane Suhr, Sergey Levine, and Aviral Kumar. Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:12461–12495, 2024.
- [6] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.
- [7] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.
- [8] Hyunjoo Chae, Namyoung Kim, Kai Tzu-iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. Web agents with world models: Learning and leveraging environment dynamics in web navigation. *arXiv preprint arXiv:2410.13232*, 2024.
- [9] Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-flan: Designing data and methods of effective agent tuning for large language models. In *Findings of the Association for Computational Linguistics (ACL)*, pp. 9354–9366, 2024.
- [10] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing gui grounding for advanced visual gui agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 9313–9332, 2024.
- [11] Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.
- [12] ModelScope Community. SWIFT: A scalable lightweight infrastructure for fine-tuning. <https://github.com/modelscope/ms-swift>, 2024. Accessed: 2025-05-15.
- [13] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- [14] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:28091–28114, 2023.
- [15] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv preprint arXiv:2401.08281*, 2024.

- [16] Google. Gemini: Try deep research and gemini 2.0 flash experimental. <https://blog.google/products/gemini/google-gemini-deep-research/>, 2024. Accessed: 2025-05-15.
- [17] Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.
- [18] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.
- [19] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pp. 6609–6625, 2020.
- [20] Jian Hu, Jason Klein Liu, and Wei Shen. Reinforce++: An efficient rlhf algorithm with robustness to both prompt and reward models. *arXiv preprint arXiv:2501.03262*, 2025. Version 3, revised 6 Apr 2025.
- [21] Yucheng Jiang, Yijia Shao, Dekun Ma, Sina Semnani, and Monica Lam. Into the unknown unknowns: Engaged human learning through participation in language model agent conversations. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9917–9955, 2024.
- [22] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-rl: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.
- [23] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.
- [24] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.
- [25] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.
- [26] Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6921–6935, 2023.
- [27] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.
- [28] Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling. *arXiv preprint arXiv:2504.02495*, 2025.
- [29] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:46534–46594, 2023.

- [30] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.
- [31] Magnus Müller and Gregor Žunič. Browser use: Enable ai to control your browser. <https://github.com/browser-use/browser-use>, 2024.
- [32] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2022.
- [33] OpenAI. Gpt-4o system card. <https://openai.com/index/gpt-4o-system-card/>, 2024. Accessed: 2025-05-15.
- [34] OpenAI. Introducing deep research, February 2025. URL <https://openai.com/index/introducing-deep-research/>.
- [35] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics (EMNLP)*, pp. 5687–5711, 2023.
- [36] Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, et al. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. *arXiv preprint arXiv:2411.02337*, 2024.
- [37] Ruslan S. 4d masks support in transformers. <https://huggingface.co/blog/poedator/4d-masks>, 2024. Hugging Face Community Blog.
- [38] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [39] Serper. Serper api: Fast and affordable google search api. <https://serper.dev/>, 2025. Accessed: 2025-05-15.
- [40] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.
- [41] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.
- [42] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:8634–8652, 2023.
- [43] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.
- [44] Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. Trial and error: Exploration-based trajectory optimization of llm agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 7584–7600, 2024.
- [45] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, second edition, 2018. URL <http://incompleteideas.net/book/the-book-2nd.html>.
- [46] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 12, pp. 1057–1063, 2000.

- [47] OpenManus Team. Openmanus: Open-source ai agent framework. <https://github.com/mannaandpoem/OpenManus>, 2025.
- [48] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pp. 5998–6008, 2017.
- [50] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(3):1–25, 2024.
- [51] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.
- [52] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 24824–24837, 2022.
- [53] Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Long-memeval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations (ICLR)*, 2025.
- [54] Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.
- [55] 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, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.
- [56] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2369–2380, 2018.
- [57] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [58] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.
- [59] Da Yin, Faeze Brahman, Abhilasha Ravichander, Khyathi Chandu, Kai-Wei Chang, Yejin Choi, and Bill Yuchen Lin. Lumos: Learning agents with unified data, modular design, and open-source llms. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2023.
- [60] Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. Compact: Compressing retrieved documents actively for question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 21424–21439, 2024.

- 527 [61] Siyu Yuan, Zehui Chen, Zhiheng Xi, Junjie Ye, Zhengyin Du, and Jiecao Chen. Agent-r: Train-
528 ing language model agents to reflect via iterative self-training. *arXiv preprint arXiv:2501.11425*,
529 2025.
- 530 [62] Zhenrui Yue, Honglei Zhuang, Aijun Bai, Kai Hui, Rolf Jagerman, Hansi Zeng, Zhen Qin,
531 Dong Wang, Xuanhui Wang, and Michael Bendersky. Inference scaling for long-context
532 retrieval augmented generation. In *Proceedings of the International Conference on Learning*
533 *Representations (ICLR)*, 2025.
- 534 [63] Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang.
535 Agenttuning: Enabling generalized agent abilities for llms. *arXiv preprint arXiv:2310.12823*,
536 2023.
- 537 [64] Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun
538 Chen, and Ningyu Zhang. Lightthinker: Thinking step-by-step compression. *arXiv preprint*
539 *arXiv:2502.15589*, 2025.
- 540 [65] Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi
541 Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. Sglang:
542 Efficient execution of structured language model programs. In *Advances in Neural Information*
543 *Processing Systems (NeurIPS)*, 2024.
- 544 [66] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and
545 Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world
546 environments. *arXiv preprint arXiv:2504.03160*, 2025.
- 547 [67] Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng,
548 Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for
549 building autonomous agents. In *Proceedings of the International Conference on Learning*
550 *Representations (ICLR)*, 2024.
- 551 [68] Banghua Zhu, Michael Jordan, and Jiantao Jiao. Principled reinforcement learning with human
552 feedback from pairwise or k-wise comparisons. In *Proceedings of the International Conference*
553 *on Machine Learning (ICML)*, pp. 43037–43067, 2023.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The abstract and introduction accurately reflect the paper's contribution and scope, with contributions bolded and highlighted in both the abstract and introduction (Sec. 1).

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We have included a discussion of the limitations in Sec. 5.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not include a theoretical result.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The code is included in the supplementary materials and will be open-sourced upon paper acceptance. In the paper, the main experiment section (Sec. 4.1) clearly describes the implementation details and hyperparameters needed to reproduce the results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code is included in the supplementary materials and will be open-sourced upon paper acceptance. All datasets used in the paper are publicly accessible via the citations provided.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All details necessary to understand the results are provided in the main experiment section (Sec. 4.1). Code is also available in the supplemental materials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Error bars that indicate the standard error of the mean are reported for all main experimental results in Tab. 1, Tab. 2, and Tab. 3.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The number and the type of GPU card used during training and evaluation are detailed in App. A.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Broader societal impacts are discussed in App. B.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: MEM1 introduces a RL framework. This research focus is on a new training methods, instead of releasing new models. This work does not create new datasets.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All sources of the datasets are cited in the paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This work does not introduce new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- 866
- 867
- 868
- 869
- 870
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
 - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

871

16. Declaration of LLM usage

872

873

874

875

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

876

Answer: [NA]

877

878

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

879

Guidelines:

- 880
- 881
- 882
- 883
- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
 - Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

884 A Details of MEM1

885 A.1 Computing Resources and Training Details

886 All trainings of MEM1 are conducted on 4 H100 or H200 GPUs. We use the veRL framework [41]
887 for RL and Swift [12] for SFT. For RL, we set the data batch size to 256 and the mini batch size to 64.
888 Learning rate is set to 10^{-6} for the actor model and 10^{-5} for the critic model with a linear warmup
889 of 50 steps. Temperature is set to 1 during training and 0.01 during inference.

890 All evaluations are conducted on one single H200 GPU, which serves respective models as an API
891 service using the vLLM framework [25] with automatic prefix caching enabled.

892 A.2 RAG Configuration

893 For RAG on local Wiki corpus, we use Faiss-GPU [15] serving a E5 Base model [51]. The Wiki
894 corpus is taken from a Wikipedia 2018 dump [23]. The number of passages for each retrieval is set to
895 3 for a fair comparison with other methods.

896 For online web search queries, we use Serper API [39], which offers Google search results including
897 titles, snippets, and URLs. For each search, we return the top 10 results to the agent as external
898 information. We do not ask the agent to retrieve the content of specific webpages.

899 A.3 Prompts

Prompt 1: Multi-Objective Task (QA)

You will answer multiple complex questions using iterative reasoning, summarization, and web search.

At each step, you will see the questions, a cumulative summary of relevant information, the current search query, and search results (except in the first step, where only the questions are provided). Your task is to:

1. Perform reasoning and update a cumulative, concise summary within `<think> ... </think>`. This acts as persistent memory and must include all essential information from previous `<think>` and `<information>` tags.
2. Then choose one of the following actions:
 - If any question remains unanswered, issue a single query for one question inside `<search> ... </search>`. The query should consist of keywords or a short phrase. Only search one question at a time.
 - If all questions are answered, provide the final answers-separated by semicolons-within `<answer> answer1; answer2; ... </answer>`. The answers must be concise, contain only essential words, and avoid any explanations.

Important:

- Always follow this structure after `<information>` or the initial questions: `<think> ... </think><search> ... </search>` or `<think> ... </think><answer> ... </answer>`.
- Do not search multiple queries or questions simultaneously.

Answer the following questions:[QUESTIONS]

900

Prompt 2: Single-Objective Task (QA)

You will answer a complex question through iterative reasoning, summarization, and web searches.

At each step, you can see the question, previous summary in `<think> ... </think>`, search query in `<search> ... </search>`, and the returned information in `<information> ... </information>` (except the first step where you will be given only the question). Then, you should:

1. Conduct reasoning, and then update a concise, cumulative summary with essential information inside `<think> </think>`. This is your persistent memory and should include all important information from previous `<think> </think>` and `<information> </information>` (i.e. information and answers already found for questions).
2. Then choose one:
 - Issue a query (i.e., key words / phrases for search) inside `<search> </search>` (you may search repeatedly until the answer is clear). This query will be used to conduct search and return the results in `<information> results </information>`
 - Provide the final concise answer (no explanations) if no additional information is needed inside `<answer> </answer>`. The answer should be concise and only contain the words necessary to answer the question.

After `<information> </information>` (or question at the beginning), you should always follow the order: `<think> ... </think><search> ... </search>` or `<think> ... </think><answer> ... </answer>`.

Question: [QUESTION]

901

Prompt 3: Single-Objective Task (WebShop)

You are browsing an online shop. Your goal is to find a product that matches the given description. You will interact with the site step-by-step. Each step gives you a `<state>...</state>` representing the current webpage. You must decide what action to take next until you identify the correct product.

Available actions (shown in the `<state>` tag) depend on the page:

- On the search page: `search[<keywords>]`
- On search result pages: `click[<item url>]` to view a product, or `click[next >]` to go to the next results page
- On product pages: `click[description]`, `click[features]`, `click[color]`, `click[size]`, `click[buy now]`
- To return to search: `click[back to search]`

Example goal: "Find a ginkgo light and 20x20 pillow cover that is hand painted." Example first action: `<answer>search[ginkgo light 20x20 pillow cover hand painted]</answer>` Only respond with valid actions formatted as: `search[...]`, `click[...]`, etc.

After you navigate and find the product that best fits the user goal, you should `click[buy now]` to buy the product at the product page when the buy now button is available.

Product Description: [PRODUCT DESCRIPTION]

902

903 A.4 Implementation Details of Metrics and Baselines

904 A.4.1 Metrics

905 **Exact match.** In QA tasks, we use exact match (EM) as both the verifiable reward for the RL
906 pipeline and the evaluation metric for the final output. The final response is extracted from between
907 `<answer>` and `</answer>`. In multi-objective settings, the response should contain answers to each
908 question separated by semicolons. If the XML tags are mismatched, or if the number of provided
909 answers does not correspond to the number of questions, a score of 0 is assigned. Otherwise, 1 point
910 is credited for each correct answer. During RL training, we do not provide any other intermediate
911 rewards or format penalties, as we find that such manual interventions can interfere with the agent’s
912 learning process (see more in Sec. 4.4).

913 **F1 score.** The F1 score computes the harmonic mean between the precision p and recall r . In the
914 case of string matching, we split both the predicted answer and the ground truth. For example, if the
915 ground truth is “United States of America”, it is split into a list with lower-case words: “united”,
916 “states”, “of”, “america”. The same works for the predicted answer. Then, denote the number of
917 common words as c . Further denote the number of words in the predicted answer as l and the number
918 of words in the ground truth as g . Then, precision is calculated as $p := c/l$ and recall is calculated as
919 $r := c/g$. The F1 score is finally computed as

$$\text{F1} := 2 \times \frac{p \times r}{p + r}.$$

920 If multiple ground truths are present, the maximum of all F1 scores is chosen. For multi-objective
921 tasks, the final F1 is the sum of the F1 scores for each sub-question.

922 **Peak tokens usage.** Peak token usage is calculated as the maximum number of tokens (using
923 GPT-4o-mini tokenizer) in any single sequence throughout the agent’s entire trajectory. For fair
924 comparison in our experiments, the system prompt is excluded when computing this sequence length.
925 The peak token usage serves as a proxy for the memory requirement for inference in deployment.

926 **Dependency length.** Following [64], the dependency metric is defined as the total number of
 927 historical tokens on which each generated token effectively depends. Let T denote the total number
 928 of interaction steps. For each step $i \in [T]$, let $n_p^{(i)}$ be the number of prefix tokens and $n_o^{(i)}$ be the
 929 number of output tokens generated. The dependency metric is then calculated as

$$\text{Dependency} := \sum_{i \in [T]} \frac{(2n_o^{(i)} + n_p^{(i)}) \times n_o^{(i)}}{2}.$$

930 At a high level, this metric quantifies the cumulative computational cost associated with the generation
 931 of an output trajectory. It is important to note that in MEM1, prefix tokens from previous steps are
 932 consolidated into a new internal state, rather than being continuously accumulated. In our experiments,
 933 we ignore the tokens in the system prompt when calculating the dependency metric.

934 **Inference time.** Inference time for each trajectory is recorded as the total elapsed time required to
 935 generate the complete output trajectory. For all experiments, these measurements are conducted on a
 936 single H200 GPU, operating with 10 concurrent threads. The vLLM inference framework is utilized,
 937 with its automatic prefix caching feature enabled.

938 A.4.2 Baselines

939 **Search-R1.** As detailed in [22], the model is trained on the 1-objective task with the same dataset
 940 as MEM1. Search-R1 also uses exact match as its reward function. In comparison, MEM1 is trained
 941 exclusively on 2-objective tasks.

942 **Deep Researcher.** As detailed in [66], the model is trained on 1-objective task with a curated set
 943 from various QA datasets including HotPotQA and Natural Questions. Deep Researcher adopts the
 944 F1 score as the reward function.

945 A.5 Algorithm

946 We provide an outline of the rollout of MEM1, which actively manages its context in Alg. 1. Parts of
 947 the pseudo-code follow [22].

Algorithm 1 MEM1 Rollout

Require: Task prompt x , policy model π_θ , world model \mathcal{W} , maximum turn T

Ensure: Final response y

```
1: Initialize rollout sequence  $y \leftarrow \emptyset$ 
2: Initialize turn count  $t \leftarrow 0$ 
3: while  $t < T$  do
4:   Initialize current policy rollout sequence  $y_t \leftarrow \emptyset$ 
5:   while True do
6:     Generate response token  $y_r \sim \pi_\theta(\cdot \mid x, y + y_t)$ 
7:     Append  $y_r$  to rollout sequence  $y_t \leftarrow y_t + y_r$ 
8:     if  $(t = T - 1)$  and  $y_r \in [\text{</answer>, <eos>}]$  then
9:       break // prevent the agent from searching further
10:    else if  $y_r \in [\text{</query>, </answer>, <eos>}]$  then
11:      break
12:    end if
13:  end while
14:   $y \leftarrow y_t$  // all previous context removed.
15:  if <query> </query> detected in  $y_t$  then
16:    Extract search query  $q \leftarrow \text{Parse}(y_t, \text{<query>, </query>})$ 
17:    Retrieve environment feedback  $d \leftarrow \mathcal{W}(q)$  from local storage, Search engine, HTML, ...
18:    HINT  $\leftarrow$  You have  $\{T - t\}$  turns left.
19:    Insert  $d$  into rollout  $y \leftarrow y + \text{<info>HINT} + d\text{</info>}$ 
20:  else if <answer> </answer> detected in  $y_t$  then
21:    return final generated response  $y$ 
22:  else
23:    Mark the sample as invalid
24:  end if
25:  Increment turn count  $t \leftarrow t + 1$ 
26: end while
27: return final generated response  $y$ 
```

948 A.6 MEM1 on Webshop Training Details

949 We use the same rollout pipeline and policy update mechanism for training MEM1 on WebShop.
950 Compared to QA task, we use a tailored prompt that retains the gist of memory consolidation with
951 instructions specific to the WebShop environment, as shown in Prompt. 3. Another distinction is
952 that the WebShop environment comes with its own reward function corresponding to each state.
953 Therefore, we do not use exact match but the built-in reward function as the reward signal when
954 training in WebShop environment. The training and test split also follows the original paper [57],
955 with the first 1000 samples as the test set, the 1000th to 1500th as the val set, and the remaining as
956 the train set.

957 B Broader Impacts

958 MEM1 opens up the potential to enable more scalable, efficient, and intelligent AI agents capable of
959 sustaining log, goal-directed interactions in dynamic environments. As AI systems are increasingly
960 deployed in complex real-world tasks—such as scientific research, legal analysis, personalized
961 education, and digital customer service—models must go beyond single-turn capabilities and manage
962 evolving contexts over many steps. MEM1’s memory-consolidation mechanism allows language
963 models to maintain high performance without the growing computational and environmental costs
964 typically associated with long-context processing. By reducing inference-time memory and compute
965 demands, MEM1 paves the way for more sustainable and scalable AI deployment, making advanced
966 reasoning agents accessible to a wider range of users and institutions, including those with limited
967 resources. Moreover, MEM1’s unified framework of reasoning and context consolidation sets
968 a precedent for future research on intelligence that can learn to adapt, reflect, and summarize
969 information autonomously, inspiring more trustworthy, interpretable, and human-aligned AI systems.

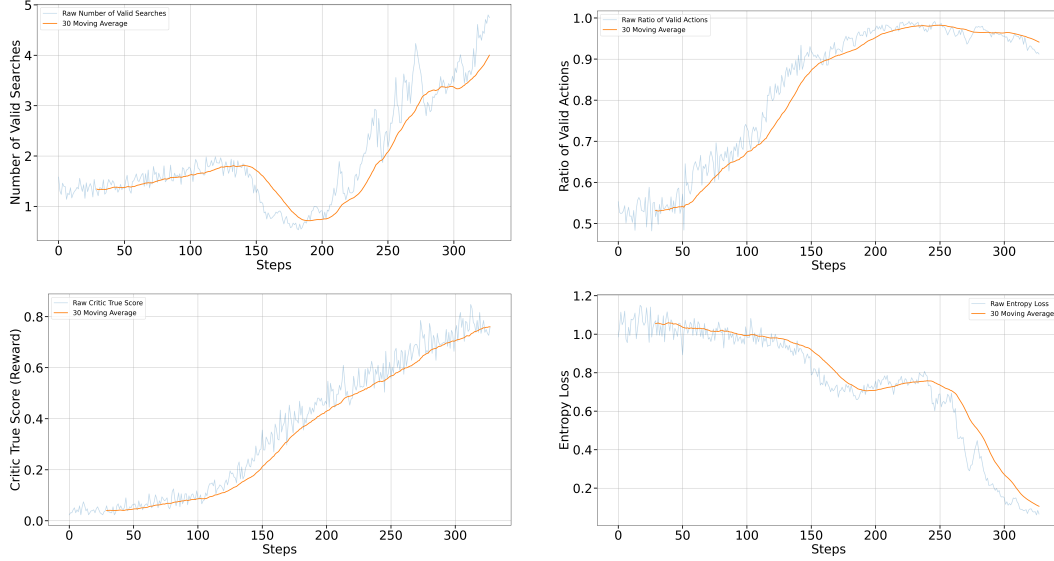


Figure 5: Metrics of training progresses for MEM1 with RL.

971 We present the training dynamics of the 2-objective QA-trained MEM1 in Fig. 5, where several
 972 distinct phases emerge during the learning process. In the initial exploration phase (first 50 steps), the
 973 agent demonstrates little task proficiency. The reward remains consistently low, while the entropy
 974 loss is high, suggesting random or undirected behavior. The ratio of valid actions hovers around 0.55,
 975 indicating that the agent frequently fails to follow the expected output format. During this period,
 976 MEM1 has not yet learned to reliably use the required structure involving `<query>` and `<answer>`
 977 tags.

978 Shortly after, we observe the onset of format acquisition. The agent gradually improves its structural
 979 consistency, reflected in the rising ratio of valid actions. This improved adherence to format correlates
 980 with an increase in reward, suggesting that proper formatting directly contributes to the agent’s task
 981 success. By around step 150, a notable behavioral shift occurs. The number of valid searches begins
 982 to drop sharply, while the reward continues to increase. This implies that the agent has discovered a
 983 shortcut: by reducing the number of searches—perhaps to avoid format violations—it can maintain
 984 high format fidelity and improve its reward without fully solving the task. This short-horizon
 985 optimization suggests the agent is exploiting the reward structure, favoring formatting compliance
 986 over content completeness.

987 Between steps 150 and 200, the agent enters a phase of refined format mastery. The ratio of valid
 988 actions steadily climbs, but the number of searches remains low. During this phase, reward growth
 989 slows, and entropy begins to flatten. The plateau in entropy indicates that the agent is looking for
 990 new policies to boost the reward. At this stage, the agent has reached a local optimum: it’s producing
 991 valid but under-informed answers.

992 After step 200, a second behavioral shift occurs. The number of valid searches begins to rise again,
 993 suggesting that the agent is learning to extend its interaction horizon to gather more information. The
 994 agent learns to balance formatting constraints with information acquisition. As a result, the reward
 995 increases more sharply. Finally, after step 250, the agent enters a phase of policy consolidation. The
 996 entropy loss drops sharply—signaling a transition from exploration to exploitation—as the agent
 997 settles into a more deterministic, high-reward policy. By this stage, the agent effectively combines
 998 format compliance, sufficient searching, and high-quality answer generation.

D Analysis on Implementation Details

D.1 RL Generalizes Better Than SFT

A natural question arises: can Supervised Fine-Tuning (SFT) with high-quality trajectories match the performance of reinforcement learning (RL)? To investigate this, we compare MEM1-QA trained via RL against MEM1-QA (SFT), where both models are trained on the 2-objective QA task. Additionally, the SFT model is further trained on 1-objective and 3-objective QA tasks to enhance its generalization ability. As shown in Tab. 4, the SFT model consistently underperforms compared to its RL counterpart across tasks with varying numbers of questions (objectives). Notably, when the number of objectives exceeds six, the performance of the SFT model collapses, whereas the RL-trained model continues to demonstrate strong robustness and scalability.

Table 4: Comparison of RL and SFT on increasing number of multi-turn questions. Exact match scores \uparrow is better. Gap shows absolute difference. Red numbers show collapsed SFT behavior.

#Q	RL \uparrow	SFT \uparrow	Gap \uparrow	RL Gain (%) \uparrow
1	0.410	0.300	0.110	+36.7%
2	0.709	0.433	0.276	+63.7%
3	0.976	0.648	0.328	+50.6%
4	1.120	0.626	0.494	+78.9%
6	1.630	0.088	1.542	+1752%
8	1.870	0.027	1.843	+6826%
16	1.900	0.000	1.900	—

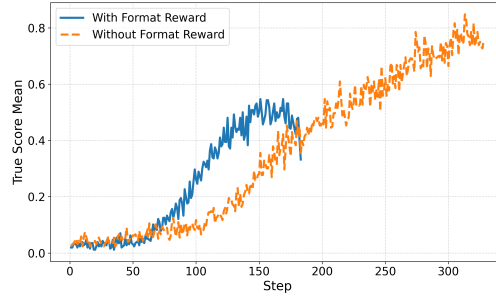


Figure 6: Training curves comparing MEM1 trained with and without format reward.

D.2 Format Reward Accelerates Convergence but Degrades Final Performance

It is common to incorporate format reward when training reasoning models and multi-turn reasoning agents [13, 66, 22]. In our study, we experimented with a format reward that enforces the agent to produce outputs using specific structural tags: `<IS>`, `<query>`, and `<answer>`. If the agent fails to use the expected tags correctly, the turn is terminated and a penalty of -1 is applied.

As shown in Fig. 6, using the format reward leads to faster convergence during training but results in worse final performance. The format-constrained agent achieves an exact match score of 0.466, compared to 0.709 for MEM1 trained with only outcome-based reward on the same testing set for the 2-objective QA task. Additionally, the format-constrained agent generates fewer tokens, with an average peak of 514.9 tokens, whereas the outcome-reward-trained MEM1 reaches an average peak of 640 tokens.

We hypothesize that the format reward accelerates structural learning but constrains exploration of effective reasoning strategies. As a result, the agent learns to produce shorter responses with valid syntax but develops less effective internal state representations, leading to degraded task performance.