MEMTRAIN: SELF-SUPERVISED CONTEXT MEMORY TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Memory is an indispensable capability for long-horizon LLM agents, enabling them to preserve and utilize information accumulated across extended interactions. Existing memory-agent approaches typically rely on task-specific end-to-end training. However, collecting high-quality annotated problems for memory-intensive scenarios is costly, and the resulting training data often lack sufficient diversity to cover general memory behaviors. In this work, we propose MemTrain, a self-supervised memory training framework for generally enhancing the context-memory capability of LLM agents. MemTrain introduces two coupled proxy tasks over unlabeled Wikipedia corpora: (1) an end-to-end masked reconstruction objective, which requires the model to recover masked entities after multiple rounds of memory updates, thereby encouraging memory maintenance from the final outcome perspective; and (2) an intermediate memory recall objective, which requires the model to reconstruct masked historical information using intermediate memory states, encouraging faithful compression and memory completeness throughout the interaction process. The two objectives are jointly optimized using GRPO. Extensive experiments on long-text QA and search-based QA benchmarks demonstrate that MemTrain significantly improves downstream memory-intensive reasoning performance across different context lengths and models, achieving gains of up to 17.67 points over direct task-specific post-training.

1 INTRODUCTION

Large language models (LLMs) have rapidly evolved into increasingly capable agents that can reason, plan, and interact with external environments (Singh et al., 2025; Team et al., 2025; DeepSeek-AI et al., 2025). However, a key bottleneck for long-horizon agentic tasks is *memory*: the ability to preserve and utilize information acquired many turns earlier. In realistic interactive settings, an agent continuously receives new observations, generates intermediate thoughts, and must maintain relevant past information across turns. A straightforward solution is to append the full interaction history to the prompt (Yao et al., 2023), but this quickly becomes prohibitively expensive as the trajectory grows. Consequently, enabling agents to operate with a *fixed-size persistent memory* remains an important challenge for scalable long-horizon deployment.

Recent work has explored *context memory* agents (Zhou et al., 2026; Yu et al., 2026; Yan et al., 2025; Yuan et al., 2026), where each interaction round is conditioned on a compact memory state rather than the entire history. As illustrated on the left of Figure 1, the agent receives $[\texttt{memory}_{t-1}; \texttt{input}_t]$, produces a response, and updates the memory to $\texttt{memory}_t$. By carrying historical information forward through a fixed-size memory, this paradigm allows near-constant context usage as the interaction proceeds and can be optimized end-to-end within the language model itself. However, information discarded during memory updates may become necessary in later turns, making effective memory maintenance and utilization essential to this paradigm.

Existing memory agents are typically trained using reinforcement learning with verifiable rewards (RLVR) on downstream tasks, but two limitations remain, as summarized in Figure 1. First, such approaches require expensive labeled data, making it difficult to obtain sufficiently diverse training data that covers the wide range of memory behaviors. Consequently, memory capabilities learned in this manner are often task-specific and exhibit limited generalization. Second, final-answer rewards provide weak supervision for memory quality. A trajectory may produce the correct answer while

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

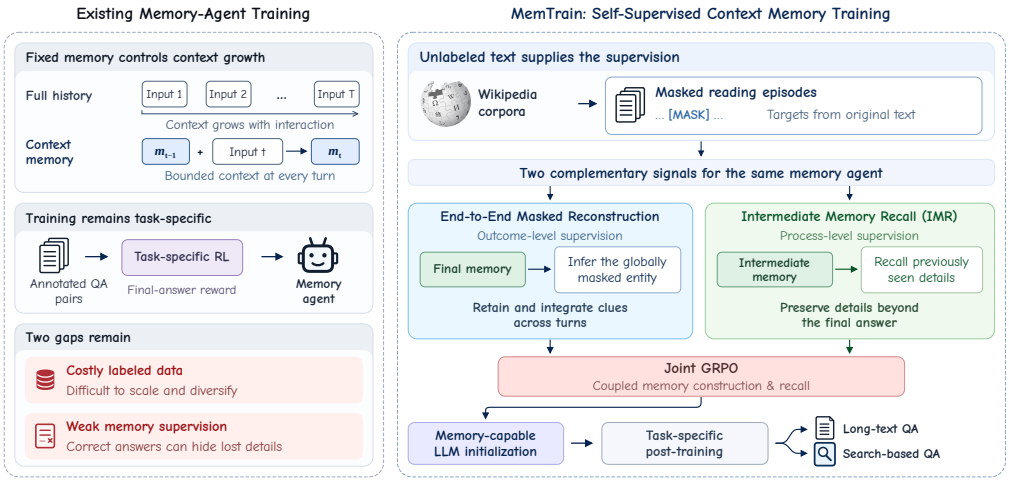

Figure 1: Background and motivation of MemTrain. **Left:** Context-memory agents compress interaction history into fixed-size memory, bounding context growth but risking information loss. Existing training relies on costly labeled data and final-answer rewards that weakly supervise memory quality. **Right:** MemTrain instead uses unlabeled Wikipedia corpora and joint GRPO to couple end-to-end masked reconstruction with intermediate memory recall, learning to retain cross-turn clues and preserve details for subsequent task-specific post-training.

retaining only the information needed for that particular problem and discarding other important historical details. Thus, successful task completion does not necessarily imply memory completeness or faithful compression throughout the interaction process. These limitations highlight the need for a task-agnostic self-supervised training paradigm that can exploit unlabeled corpora while providing richer supervision for memory generation and utilization.

Meanwhile, recent advances in reasoning have explored reinforcement learning with pre-training data (Dong et al., 2025; Li et al., 2025; Xing et al., 2025). They construct self-supervised proxy tasks over unlabeled corpora using chain-of-thought next-token prediction to generally improve reasoning ability. However, memory learning poses distinct challenges from reasoning. The memory target is inherently latent and process-dependent, as the model must continuously decide what information to preserve, compress, and recall over time. Consequently, designing a proxy task that faithfully captures the underlying memory mechanism remains a significant challenge. In particular, supervision should assess not only whether the final memory supports the desired prediction, but also whether intermediate memories preserve information that can be recalled later.

To address this challenge, we propose **MemTrain**, a self-supervised memory training framework for improving the context-memory capability of LLM agents. As illustrated on the right of Figure 1, MemTrain constructs memory training examples from unlabeled Wikipedia passages and derives supervision targets from the original text, avoiding the need to collect annotated problems for memory training. To provide complementary supervision for memory generation and utilization, we introduce two coupled proxy tasks: (1) an end-to-end masked reconstruction task, which requires the model to recover masked entities after multiple rounds of memory updates, thereby providing outcome-level supervision that encourages the model to retain and integrate clues across turns; and (2) an intermediate memory recall task, which requires the model to reconstruct additional masked entities from earlier interaction history using intermediate memory states, thereby providing process-level supervision that encourages the preservation of historical details beyond those needed for the final prediction. The two objectives are jointly optimized with GRPO, coupling memory construction and recall within the same model. MemTrain finally produces a memory-capable initialization for subsequent task-specific post-training such as long-text QA and search-based QA. Extensive experiments show that MemTrain consistently improves downstream long-text QA and search-based QA performance over direct task training. The average improvements reach 5.17 and 10.58 points, respectively, on Qwen3-4B-Instruct-2507, and 17.67 and 8.50 points, respectively, on Qwen2.5-7B-Instruct.

Our contributions are summarized as follows:

- We propose MemTrain, the first self-supervised memory training framework designed to generally improve the context-memory capability of LLM agents.
- We introduce a novel memory-oriented proxy training paradigm that jointly provides outcome-level and process-level supervision signals for memory generation and utilization.
- Extensive experiments on two models demonstrate that MemTrain consistently improves downstream performance on both long-text QA and search-based QA tasks.

2 Related Work

Memory for Long-Horizon LLM Agents. The most widely adopted memory management strategy for LLM agents is to continually append environmental observations and model responses to the context window (Yao et al., 2023), which is fundamentally limited by the finite context window of LLMs. To enable unbounded memory, external memory systems have been proposed, where interaction records are compressed or summarized and stored externally (Yoon et al., 2024; Li et al., 2023; Chhikara et al., 2025; Xu et al., 2025). Qian et al. (2026); Xu et al. (2025); Chen et al. (2026) further introduce multi-agent frameworks to support more sophisticated and efficient memory management. However, external memory systems often overlook the intrinsic synergy between memory and reasoning, while simultaneously increasing overall system complexity. More recent studies (Zhou et al., 2026; Yu et al., 2026; Wu et al., 2026; Ye et al., 2025; Yuan et al., 2026) integrate memory construction and utilization directly into the reasoning process of the agent itself, enabling end-to-end optimization. Despite their effectiveness, these approaches typically rely on costly task-specific annotations, severely limiting the data diversity. In this work, we instead propose a self-supervised memory training framework that enables training on common Internet corpora.

Reinforcement Learning for LLM Pre-training. Reinforcement learning has been extensively adopted during post-training to enhance the reasoning and tool-use capabilities of LLMs (DeepSeek-AI et al., 2025; Yu et al., 2025). However, post-training methods generally depend on curated question-answer datasets, which limits both scalability and generalization. Motivated by the success of self-supervised language model pre-training, recent works have explored reinforcement pre-training paradigms that leverage large-scale Internet text. Quiet-STaR (Zelikman et al., 2024; Huang et al., 2025) generates latent rationales at each token position to better predict future text. RPT (Dong et al., 2025) introduces the next-token reasoning RLVR objective and demonstrates scalable reinforcement learning pre-training for the first time. RLPT (Li et al., 2025) adopts a similar formulation while incorporating a generative reward model. RLP (Hatamizadeh et al., 2025) replaces next-token prediction with a contrastive reward to explicitly induce reasoning. PretrainZero (Xing et al., 2025) further proposes an active pre-training framework that synthesizes more informative and valuable training samples. Nevertheless, existing RL-based pre-training approaches primarily focus on single-turn reasoning, leaving the problem of learning effective multi-turn memory maintenance and utilization largely unexplored.

3 Memory Training

In this section, we first formulate the context memory agent (§ 3.1). We then introduce the two memory training tasks – end-to-end masked reconstruction (§ 3.2) and intermediate memory recall (§ 3.3). Finally, we describe how we conduct the memory training using GRPO (§ 3.4).

3.1 Problem Setup

Our study is built upon the framework of multi-turn context memory proposed in MemAgent (Yu et al., 2026). The context-memory schematic on the left of Figure 1 illustrates its central mechanism: historical information is carried forward in a fixed-size memory and combined with the current input at each turn. We formalize this mechanism using a memory state m_t at interaction step t. At each step, the model receives an input tuple (m_{t-1}, a_{t-1}, i_t), where a_{t-1} is the preceding action and i_t is the current input or environment feedback. Conditioned on this tuple, the model produces the

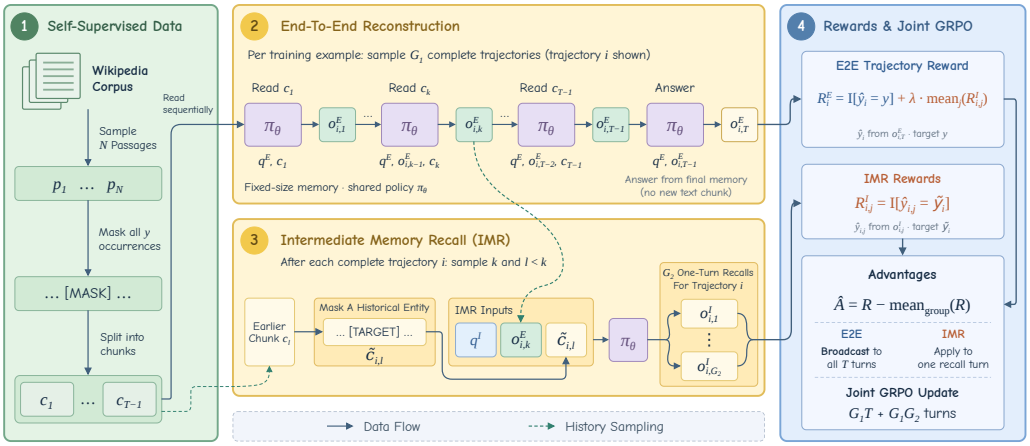

Figure 2: Overview of MemTrain. (1) We sample N Wikipedia passages, mask a target entity y, and split the text into chunks $c_{1:T-1}$. (2) We sample G_1 trajectories that sequentially read the chunks, update context memory, and predict the masked entity. (3) For each trajectory, we sample an intermediate memory and an earlier chunk, then generate G_2 one-turn recalls to recover an entity masked in that chunk. (4) We couple reconstruction and recall rewards, broadcast each E2E advantage across its T turns, and jointly optimize all $G_1T + G_1G_2$ turns with GRPO.

updated memory state and the current action, i.e., (m_t, a_t), which are then used in the subsequent interaction step.

The action space depends on the target application. For long-context reading agents, actions may correspond to requesting the next text chunk or generating the final answer. For search agents, actions may involve invoking an external search tool or directly returning an answer. When the preceding action interacts with the environment, i_t represents the input or feedback returned after executing a_{t-1}. This formulation instantiates the shared memory-update mechanism in Figure 1 for both long-context reading and search-based interaction.

Compared with the conventional agent paradigm, where the entire interaction history is continually appended to the context window, context memory maintains a constant context size throughout the trajectory. This design removes the dependence on ever-growing context length, enabling long-horizon interaction beyond the model's native context limit while mitigating attention dilution and avoiding the increasing computational cost associated with long-context processing.

3.2 End-to-End Masked Reconstruction

We construct training samples from raw Wikipedia text. First, we randomly select one passage as the pivot passage. We then retrieve n_1 semantically related passages from the corpus together with $N-n_1-1$ randomly sampled passages. These N passages are concatenated in random order to form a long document. Next, we randomly select an entity y (e.g., a number or location) from the pivot passage and replace all occurrences of this entity in the document with a special token [MASK].

Following the practice in context-memory research (Yu et al., 2026), we segment the long document into fixed-length chunks $\{c_1, c_2, \ldots, c_{T-1}\}$, where each chunk corresponds to an interaction step. The LLM sequentially processes these chunks to generate a multi-turn trajectory o_i^E (the i-th rollout) following $o_{i,t}^E \sim \pi_\theta(\cdot|q^E, x_{i,t}^E)$, where $x_{i,t}^E = o_{i,t-1}^E \oplus c_t$ and q^E denotes the reconstruction prompt detailed in Appendix A. For $t < T$, the output $o_{i,t}^E$ serves as the context memory for the next interaction step, while $o_{i,T}^E$ denotes the final answer prediction generated solely based on the memory state $o_{i,T-1}^E$, without external input. Since all occurrences of y are masked, the model cannot simply copy the answer from the document and must instead infer the masked entity through comprehensive long-range information aggregation. This setup provides an end-to-end supervision signal: successful prediction requires preserving and integrating relevant information across multiple memory updates rather than relying on local context alone.

3.3 Intermediate Memory Recall

End-to-end rewards alone are often coarse and may not sufficiently constrain the quality of intermediate memory states. The model may incidentally preserve the information necessary for the final prediction while discarding other important details. Furthermore, due to error accumulation across multiple interaction steps, optimization based solely on end-to-end outcomes may provide weak and unstable learning signals.

To address this issue, we introduce the Intermediate Memory Recall (IMR) task. After generating the i-th complete trajectory o_i^E, we randomly select an intermediate interaction step k. We then take the corresponding memory state $o_{i,k}^E$ together with a randomly selected previous chunk input c_l ($l < k$). The model is then required to recover the entity $\tilde{y}_i$ from the masked chunk $\tilde{c}_l$ within a single interaction step, following $o_{i,j}^I \sim \pi_\theta(\cdot|q^I, \tilde{x}_i)$, where $\tilde{x}_i = o_{i,k}^E \oplus \tilde{c}_l$ and q^I is the IMR task prompt detailed in Appendix A.

This objective explicitly encourages the model to preserve sufficient historical information within the current memory state. As a result, the learned memory representations become both information-rich and directly retrievable for downstream reasoning.

3.4 Joint GRPO Optimization

We employ GRPO as the reinforcement learning algorithm. Figure 2 provides an overview. For each training sample $(p_{1:N}, y)$, we first sample G_1 end-to-end trajectories $\{o_i^E\}_{i=1}^{G_1}$ under the current policy. Then, for each sampled trajectory o_i^E, we construct one IMR prompt and further sample G_2 IMR trajectories $\{o_{i,j}^I\}_{j=1}^{G_2}$. We extract the answers $\hat{y}_i^E$ and $\hat{y}_{i,j}^I$ from these trajectories and compute the exact-match reward. For the IMR task, we have:

$$R_{i,j}^I = \mathbb{I}[\hat{y}_{i,j}^I = \tilde{y}_i]. \tag{1}$$

For the end-to-end task, the reward consists of two components: the exact-match reward for the final prediction and the associated IMR rewards:

$$R_i^E = \mathbb{I}[\hat{y}_i^E = y] + \frac{\lambda}{G_2} \sum_{j=1}^{G_2} R_{i,j}^I, \tag{2}$$

where λ is a balancing coefficient. The intuition behind this design is twofold. First, IMR rewards directly train the model to retrieve and reason over information stored in memory. Second, augmenting end-to-end rewards with IMR outcomes encourages the model to generate memory states that remain useful for future retrieval and reasoning.

Since each end-to-end trajectory consists of multiple interaction steps, we treat each step as an independent conversation instance for advantage estimation and policy optimization. Following Dr. GRPO (Liu et al., 2025), we adopt the unnormalized advantage formulation:

$$\hat{A}_{i,j,k} = \begin{cases} R_i^E - \dfrac{1}{G_1} \displaystyle\sum_{i'=1}^{G_1} R_{i'}^E, & j \le |o_i^E|, \\ R_{i,j-|o_i^E|}^I - \dfrac{1}{G_2} \displaystyle\sum_{j'=1}^{G_2} R_{i,j'}^I, & j > |o_i^E|. \end{cases} \tag{3}$$

where i, j, and k denote the indices of the trajectory, interaction step, and token, respectively. The advantage computed from the final trajectory reward is broadcast to all interaction steps. Finally, all end-to-end and IMR samples are jointly optimized using the GRPO objective in Eq. (4). For notational simplicity, we omit $q^{E/I}$ and define a unified trajectory collection $o_i = (o_{i,1}^E, \cdots, o_{i,|o_i^E|}^E, o_{i,1}^I, \cdots, o_{i,G_2}^I)$, which combines the end-to-end trajectory with its associated IMR trajectories.

$$\mathcal{J}(\theta) = \mathbb{E}_{(p,y)\sim\mathcal{D}, \{o_i^E\}_{i=1}^{G_1}\sim\pi_\theta(\cdot|c), \{o_{i,j}^I\}_{j=1}^{G_2}\sim\pi_\theta(\cdot|\tilde{x}_i)} \left[\frac{1}{\sum_{i=1}^{G_1} |o_i^E| + G_1 G_2} \sum_{i=1}^{G_1} \sum_{j=1}^{|o_i|} \sum_{k=1}^{|o_{i,j}|} C_{i,j,k} \right], \tag{4}$$

Model	Length								
	7k	14k	28k	56k	112k	224k	448k	896k	Avg
Qwen3-4B-Instruct	57.81	51.56	34.38	10.94	8.59	4.69	3.91	3.91	21.97
+MemTrain	63.28	60.16	60.16	57.03	60.94	58.59	48.44	40.62	56.15
+MemAgent	70.31	64.06	71.88	62.50	64.84	66.41	**64.06**	57.03	65.14
+MemTrain+MemAgent	**79.69**	**73.44**	**75.78**	**73.44**	**68.75**	**67.19**	61.72	**62.50**	**70.31**
Qwen2.5-7B-Instruct	53.12	51.56	35.16	13.28	10.16	1.56	1.56	0.00	20.80
+MemTrain	59.38	55.47	48.44	46.09	42.19	38.28	39.84	33.59	45.41
+MemAgent	64.06	67.19	62.50	59.38	55.47	50.00	46.88	41.41	55.86
+MemTrain+MemAgent	**76.56**	**79.69**	**77.34**	**75.00**	**70.31**	**75.78**	**64.84**	**68.75**	**73.53**

Table 1: Model performance for long-text QA across different context lengths.

$$C_{i,j,k} = \min\Big(r_{i,j,k}(\theta)\hat{A}_{i,j,k}, \mathrm{clip}(r_{i,j,k}(\theta), 1-\varepsilon_{\mathrm{low}}, 1+\varepsilon_{\mathrm{high}})\hat{A}_{i,j,k}\Big) - \beta D_{\mathrm{KL}}(\pi_\theta||\pi_{\mathrm{ref}}),$$

$$r_{i,j,k}(\theta) = \begin{cases} \frac{\pi_\theta(o_{i,j,k}|x^E_{i,j}, o_{i,j,<k})}{\pi_{\mathrm{old}}(o_{i,j,k}|x^E_{i,j}, o_{i,j,<k})} & j \le |o^E_i|, \\ \frac{\pi_\theta(o_{i,j,k}|\tilde{x}_i, o_{i,j,<k})}{\pi_{\mathrm{old}}(o_{i,j,k}|\tilde{x}_i, o_{i,j,<k})} & j > |o^E_i|. \end{cases}$$

4 Experiments

We evaluate the effectiveness of MemTrain by measuring the final downstream performance after post-training. We consider two representative tasks: (1) long-context multi-hop question answering (§ 4.2), which closely matches the memory training setting where the model reads chunked long documents and answers questions; and (2) multi-hop question answering with search tools (§ 4.3), an **out-of-domain** retrieval-augmented setting in which the model iteratively retrieves external information and performs reasoning to produce the final answer. For post-training, we adopt MemAgent (Yu et al., 2026) and MEM1 (Zhou et al., 2026), as they are the only open-source algorithms among related works.

4.1 Memory Training Setup

Dataset. We use Wikipedia as a general-purpose unsupervised corpus for memory training. Entities are identified using the NER system provided by the spaCy library. For each pivot passage, we retrieve the top-29 semantically related passages from the corpus and further augment them with 120 randomly sampled passages. This process produces 30k training documents with lengths ranging from 24k to 40k tokens.

Implementation. Our training framework is implemented using veRL (Sheng et al., 2025). We adopt GRPO (DeepSeek-AI et al., 2025) with a KL regularization coefficient of 1×10^{-3}, and follow DAPO (Yu et al., 2025) by filtering out samples whose rewards are entirely zero or entirely one. Following prior context memory agent works (Yu et al., 2026; Zhou et al., 2026), we limit the context length to 8192 tokens, including 1024 tokens for instructions, 5120 tokens for input chunks, 1024 tokens for memory, and 1024 tokens for model responses. Consequently, each input consists of at most $40k/5k = 8$ chunks. We use a batch size of 32, generate $G_1 = 8$ end-to-end rollouts, and sample $G_2 = 8$ IMR trajectories for each rollout. Training is conducted for 300 steps with a learning rate of 1×10^{-6}. The IMR coefficient λ is set to 0.5. For backbone model selection, we evaluate two widely used instruction-tuned models: Qwen3-4B-Instruct-2507 and Qwen2.5-7B-Instruct.

4.2 Long-Text Multi-Hop QA

Post-Training. We adopt MemAgent (Yu et al., 2026) as the downstream post-training algorithm. All hyperparameters follow the settings described in the MemAgent paper. We train for 500 steps until convergence using a rollout batch size of 32, an update batch size of 8, and a learning rate of 1×10^{-6}. For each backbone, we train two variants: one directly post-trained with MemAgent and another initialized from the MemTrain checkpoint before post-training, with three different seeds.

Model	TriviaQA	Bamboogle	HotpotQA	NQ	PopQA	2Wiki	MuSiQue	Avg
Qwen3-4B-Instruct-2507	42.71	21.78	18.94	19.92	21.81	14.36	4.76	20.61
+MEM1	44.29	23.39	18.80	21.97	23.62	12.80	5.63	21.50
+MemTrain+MEM1	**55.63**	**34.68**	**27.85**	**32.24**	**37.91**	**25.84**	**10.43**	**32.08**
Qwen2.5-7B-Instruct	18.84	8.87	11.15	12.22	12.59	10.45	4.43	11.22
+MEM1	49.08	22.58	19.79	24.21	27.13	17.81	6.96	23.94
+MemTrain+MEM1	**57.21**	**30.65**	**27.73**	**35.18**	**38.36**	**27.32**	**10.64**	**32.44**

Table 2: Model performance for multi-hop QA with search tools across different benchmarks.

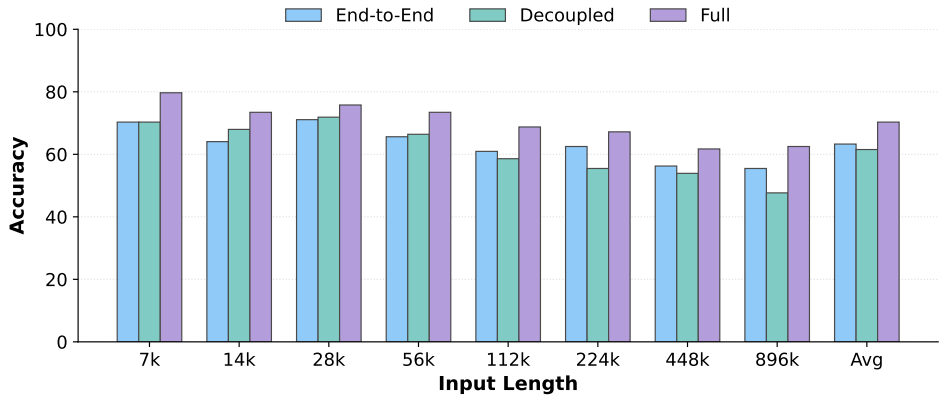

Figure 3: Ablation results using Qwen3-4B-Instruct-2507 on long-context HotpotQA.

Evaluation. We evaluate on the long-context HotpotQA benchmark introduced by Yu et al. (2026), which is specifically designed to study performance under varying context lengths. The input length ranges from 7k to 896k tokens. For direct evaluation of the original backbone models, the entire document is provided in a single context window. For models trained with MemTrain or MemAgent, we adopt the chunked memory pipeline.

Results. Table 1 demonstrates that our memory training framework consistently provides substantial gains for subsequent memory-oriented post-training. Compared with directly applying MemAgent, the combination of MemTrain and MemAgent achieves significantly higher average performance on both backbone models, improving by 5.17 percentage points on Qwen3-4B-Instruct and 17.67 percentage points on Qwen2.5-7B-Instruct. More importantly, these improvements are highly consistent across all context lengths, ranging from 7k to 896k tokens, indicating that the proposed memory training provides a strong initialization for downstream long-horizon memory learning.

Another notable observation is the strong length generalization ability introduced by MemTrain. Although the training context length (24k$\sim$40k) is closest to 28k, the gains transfer effectively to both substantially shorter and longer contexts. This effect is particularly evident on Qwen2.5-7B-Instruct. While MemAgent drops from 62.50% at 28k to 41.41% at 896k, corresponding to a decrease of 21.09 points, MemTrain+MemAgent only decreases from 77.34% to 68.75%, a much smaller drop of 8.59 points despite the 32$\times$ increase in context length. The improvements also extend to shorter contexts such as 7k and 14k, indicating that MemTrain learns more transferable and length-generalizable memory maintenance and retrieval behaviors rather than overfitting to a specific training horizon. Similar trends are consistently observed on Qwen3-4B-Instruct.

Furthermore, MemTrain alone already endows the model with considerable multi-turn question answering and memory capabilities, despite being trained entirely without labeled supervision. Compared with the original models, MemTrain improves the average performance from 21.97% to 56.15% on Qwen3-4B-Instruct and from 20.80% to 45.41% on Qwen2.5-7B-Instruct.

4.3 Multi-Hop QA with Search Tools

Post-Training. We adopt MEM1 (Zhou et al., 2026) as the downstream post-training algorithm. Following the original MEM1 setup, training is performed on 2-objective HotpotQA and Natural Questions, with at most 6 search turns and a length limit of 1k tokens for both model responses and

MemAgent	MemTrain+MemAgent
Question	
The director of the romantic comedy "Big Stone Gap" is based in what New York city?	
Chunk 1	
(Documents 1–36 omitted) ...Document 37: Adriana Trigiani is an Italian American best-selling author, television writer, film director, and entrepreneur based in Greenwich Village, New York City.	
Memory 1 ...The problem asks about the director of "Big Stone Gap." However, after reviewing the provided section, **no film titled "Big Stone Gap" is mentioned**. The only romantic comedy films discussed are "Nola" and "Just Another Romantic Wrestling Comedy"...	**Memory 1** ...Document 37 mentions Adriana Trigiani, an Italian American best-selling author, television writer, **film director**, and entrepreneur **based in Greenwich Village, New York City**. While she is a film director and based in New York City, there is no direct evidence that she directed "Big Stone Gap"...
Chunk 2	
(Documents 38–43 omitted) ...Document 44: *Big Stone Gap* is a 2014 American drama romantic comedy film written and directed by Adriana Trigiani and produced by Donna Gigliotti... (Documents 45–50 omitted)	
Memory 2 ...Document 44 confirms "Big Stone Gap" (2014) was written and directed by Adriana Trigiani. **However, no information about her base location is available in memory.** Thus, the required information is not present...	**Memory 2** ...Document 44 confirms "Big Stone Gap" (2014) was **written and directed by Adriana Trigiani**. Document 37 confirms Adriana Trigiani is **based in Greenwich Village, New York City**. Thus, the director is based in ...
...	
Final Answer **Cannot be determined**	**Final Answer** **Greenwich Village**

Table 3: Case study comparing models with and without MemTrain. Key information is underlined in the input chunks and highlighted in bold within the memory. Critical differences are marked in red and blue. We observe that MemTrain helps MemAgent keep more information in memory.

retrieved search results. We employ the same retriever and local database as MEM1, and train for 200 steps until convergence using a rollout batch size of 32, an update batch size of 8, and a learning rate of 5×10^{-7}. As in the long-context QA setting, we train both a directly post-trained model and a model initialized from MemTrain.

Evaluation. We evaluate on 7 challenging multi-hop QA benchmarks, including 2WikiMultiHopQA (Ho et al., 2020), Bamboogle (Mallen et al., 2023), HotpotQA (Yang et al., 2018), TriviaQA (Joshi et al., 2017), Natural Questions, PopQA, and MuSiQue (Trivedi et al., 2022). Following the MEM1 implementation, we augment the evaluation set into a two-objective setting and report exact-match accuracy averaged across the two objectives.

Results. Table 2 shows that MemTrain generalizes well to search-based multi-hop QA despite a clear distribution shift from memory training. Across models, MemTrain+MEM1 consistently improves over MEM1 on all benchmarks. On Qwen3-4B-Instruct-2507, the average performance increases by 10.58 points, and on Qwen2.5-7B-Instruct by 8.50 points. MemTrain-only models are not included in the comparison because they have not been exposed to a tool-use environment.

The improvements are consistent across datasets and are more pronounced on harder multi-hop tasks. In particular, the largest gains are observed on PopQA, NQ, and 2Wiki, with improvements of +11.23, +10.97, and +9.51 on Qwen2.5-7B-Instruct, and +14.29, +10.27, and +13.04 on Qwen3-4B-Instruct-2507, respectively. This may be attributed to the fact that these tasks require maintaining and integrating a larger number of pieces of intermediate evidence across retrieval steps, where improved memory construction and utilization from memory training becomes more critical. Notably, on MuSiQue, directly applying MEM1 yields only marginal improvements over the base model (e.g., +2.53 on Qwen2.5-7B-Instruct), whereas incorporating MemTrain leads to a much larger gain (+6.21), suggesting that memory training is particularly beneficial in retrieval-sensitive settings.

5 Analysis

5.1 Ablation Study

To further investigate the contribution of each component in MemTrain, we design two ablation variants: (1) **End-to-End**, which removes the IMR branch and retains only the end-to-end prediction

objective; and (2) **Decoupled**, which computes rewards for end-to-end trajectories solely based on final prediction correctness, decoupled from IMR.

As shown in Figure 3, the Full model consistently outperforms both ablation variants across all evaluated context lengths, demonstrating the importance of IMR. Specifically, removing the IMR branch decreases the average score from 70.31% to 63.28%. This degradation consistently appears across all context lengths, indicating that the end-to-end prediction objective alone does not provide sufficient supervision for identifying and preserving critical information throughout extremely long interaction histories.

Compared with the End-to-End variant, the Decoupled variant achieves stronger performance on relatively shorter contexts ($\leq 56k$), suggesting that IMR learning improves memory utilization. However, its performance deteriorates significantly as the context length increases. One possible explanation is that the decoupled objective fails to provide sufficient guidance for high-quality memory generation, forcing the model to solve tasks based on poorly constructed memories and consequently leading to more severe hallucination under long-horizon settings.

5.2 Memory Training vs. Post-Training Scaling

In this section, we compare the gains brought by memory training with those obtained from simply scaling post-training. Starting from the MemAgent checkpoint at step 500 on Qwen3-4B-Instruct-2507, we continue post-training for an additional 300 steps. We report the average accuracy across all input lengths.

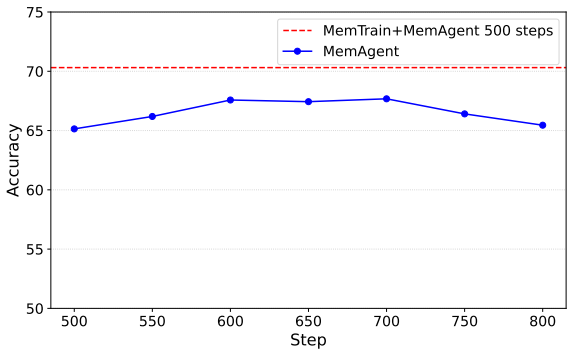

Figure 4: Performance comparison between MemTrain and continual post-training.

As shown in Figure 4, post-training is already close to saturation after step 500, and further scaling yields only marginal improvements or even performance degradation. Even at the best-performing checkpoint around step 700, the model initialized with MemTrain still maintains an advantage of 2.64 percentage points. These results suggest that although memory training introduces additional computational cost, it effectively raises the performance ceiling of downstream post-training in a manner that cannot be replicated by simply extending post-training. Therefore, allocating additional GPU resources to memory training appears to be a meaningful investment.

5.3 Case Study

We present a representative case of Qwen3-4B-Instruct-2507 to understand the effect of MemTrain. As shown in Table 3, direct MemAgent fails to retain the critical information at the memory update step after chunk 1, resulting in an inability to answer despite finding the director's identity in chunk 2. MemTrain successfully preserves the key entity information (Adriana Trigiani's location) in memory from chunk 1, enabling correct answer deduction in chunk 2.

6 Conclusion

In this work, we introduce MemTrain, the first self-supervised framework for improving the general-purpose memory capability of LLMs. We design two coupled proxy tasks—end-to-end masked reconstruction and intermediate memory recall—to jointly encourage memory completeness, faithful compression, and effective utilization. We perform memory training on Wikipedia corpora and demonstrate consistent improvements on downstream long-text and search-based question answering tasks across two models.

AI USE STATEMENT

We only use large language models for grammar checking and expression polishing, without involving any research uses such as finding related work, research ideation, and experimental design. All research concepts, ideas, and analyses were developed and conducted by the authors. We provide the original manuscript to OpenAI GPT-5 Sol and instruct it to *check the errors and make the given text more professional, coherent, and native as part of a research paper*.

We have ensured that the LLM-generated text adheres to ethical guidelines and does not contribute to plagiarism or scientific misconduct. All generated text is carefully reviewed to confirm factual accuracy and fidelity to the original content.

REPRODUCIBILITY STATEMENT

The MemTrain algorithm is described in Section 3. We also provide a detailed pseudocode in Appendix B. The experimental setup is mainly described in Section 4.1 and the prompt templates used for RL training are provided in Appendix A. To facilitate reproduction, we release the code at https://anonymous.4open.science/r/MemTrain-DB2A.

REFERENCES

Rubing Chen, Jian Wang, Wenjie Li, Xiao-Yong Wei, and Qing Li. To Retrieve or To Think? An Agentic Approach for Context Evolution, January 2026.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory. In Inês Lynce, Nello Murano, Mauro Vallati, Serena Villata, Federico Chesani, Michela Milano, Andrea Omicini, and Mehdi Dastani (eds.), *ECAI 2025 - 28th European Conference on Artificial Intelligence, 25-30 October 2025, Bologna, Italy - Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025)*, Frontiers in Artificial Intelligence and Applications, pp. 2993–3000. IOS Press, 2025. doi: 10.3233/FAIA251160.

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, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu

Zhang, and Zhen Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, January 2025.

Qingxiu Dong, Li Dong, Yao Tang, Tianzhu Ye, Yutao Sun, Zhifang Sui, and Furu Wei. Reinforcement Pre-Training, June 2025.

Ali Hatamizadeh, Syeda Nahida Akter, Shrimai Prabhumoye, Jan Kautz, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, and Yejin Choi. RLP: Reinforcement as a Pretraining Objective. In *The Fourteenth International Conference on Learning Representations*, October 2025.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580.

Wei Huang, Yizhe Xiong, Xin Ye, Zhijie Deng, Hui Chen, Zijia Lin, and Guiguang Ding. Fast Quiet-STaR: Thinking Without Thought Tokens. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 18771–18781, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.1020.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147.

Siheng Li, Kejiao Li, Zenan Xu, Guanhua Huang, Evander Yang, Kun Li, Haoyuan Wu, Jiajia Wu, Zihao Zheng, Chenchen Zhang, Kun Shi, Kyrierl Deng, Qi Yi, Ruibin Xiong, Tingqiang Xu, Yuhao Jiang, Jianfeng Yan, Yuyuan Zeng, Guanghui Xu, Jinbao Xue, Zhijiang Xu, Zheng Fang, Shuai Li, Qibin Liu, Xiaoxue Li, Zhuoyu Li, Yangyu Tao, Fei Gao, Cheng Jiang, Bo Chao Wang, Kai Liu, Jianchen Zhu, Wai Lam, Wayyt Wang, Bo Zhou, and Di Wang. Reinforcement Learning on Pre-Training Data, September 2025.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. Compressing Context to Enhance Inference Efficiency of Large Language Models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6342–6353, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.391.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-Like Training: A Critical Perspective, March 2025.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9802–9822, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.546.

Hongjin Qian, Zhao Cao, and Zheng Liu. MemoBrain: Executive Memory as an Agentic Brain for Reasoning, January 2026.

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. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, March 2025. doi: 10.1145/3689031.3696075.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, A. J. Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex

Beutel, Alex Karpenko, Alex Makelov, Alex Neitz, Alex Wei, Alexandra Barr, Alexandre Kirchmeyer, Alexey Ivanov, Alexi Christakis, Alistair Gillespie, Allison Tam, Ally Bennett, Alvin Wan, Alyssa Huang, Amy McDonald Sandjideh, Amy Yang, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrei Gheorghe, Andres Garcia Garcia, Andrew Braunstein, Andrew Liu, Andrew Schmidt, Andrey Mereskin, Andrey Mishchenko, Andy Applebaum, Andy Rogerson, Ann Rajan, Annie Wei, Anoop Kotha, Anubha Srivastava, Anushree Agrawal, Arun Vijayvergiya, Ashley Tyra, Ashvin Nair, Avi Nayak, Ben Eggers, Bessie Ji, Beth Hoover, Bill Chen, Blair Chen, Boaz Barak, Borys Minaiev, Botao Hao, Bowen Baker, Brad Lightcap, Brandon McKinzie, Brandon Wang, Brendan Quinn, Brian Fioca, Brian Hsu, Brian Yang, Brian Yu, Brian Zhang, Brittany Brenner, Callie Riggins Zetino, Cameron Raymond, Camillo Lugaresi, Carolina Paz, Cary Hudson, Cedric Whitney, Chak Li, Charles Chen, Charlotte Cole, Chelsea Voss, Chen Ding, Chen Shen, Chengdu Huang, Chris Colby, Chris Hallacy, Chris Koch, Chris Lu, Christina Kaplan, Christina Kim, C. J. Minott-Henriques, Cliff Frey, Cody Yu, Coley Czarnecki, Colin Reid, Colin Wei, Cory Decareaux, Cristina Scheau, Cyril Zhang, Cyrus Forbes, Da Tang, Dakota Goldberg, Dan Roberts, Dana Palmie, Daniel Kappler, Daniel Levine, Daniel Wright, Dave Leo, David Lin, David Robinson, Declan Grabb, Derek Chen, Derek Lim, Derek Salama, Dibya Bhattacharjee, Dimitris Tsipras, Dinghua Li, Dingli Yu, D. J. Strouse, Drew Williams, Dylan Hunn, Ed Bayes, Edwin Arbus, Ekin Akyurek, Elaine Ya Le, Elana Widmann, Eli Yani, Elizabeth Proehl, Enis Sert, Enoch Cheung, Eri Schwartz, Eric Han, Eric Jiang, Eric Mitchell, Eric Sigler, Eric Wallace, Erik Ritter, Erin Kavanaugh, Evan Mays, Evgenii Nikishin, Fangyuan Li, Felipe Petroski Such, Filipe de Avila Belbute Peres, Filippo Raso, Florent Bekerman, Foivos Tsimpourlas, Fotis Chantzis, Francis Song, Francis Zhang, Gaby Raila, Garrett McGrath, Gary Briggs, Gary Yang, Giambattista Parascandolo, Gildas Chabot, Grace Kim, Grace Zhao, Gregory Valiant, Guillaume Leclerc, Hadi Salman, Hanson Wang, Hao Sheng, Haoming Jiang, Haoyu Wang, Haozhun Jin, Harshit Sikchi, Heather Schmidt, Henry Aspegren, Honglin Chen, Huida Qiu, Hunter Lightman, Ian Covert, Ian Kivlichan, Ian Silber, Ian Sohl, Ibrahim Hammoud, Ignasi Clavera, Ikai Lan, Ilge Akkaya, Ilya Kostrikov, Irina Kofman, Isak Etinger, Ishaan Singal, Jackie Hehir, Jacob Huh, Jacqueline Pan, Jake Wilczynski, Jakub Pachocki, James Lee, James Quinn, Jamie Kiros, Janvi Kalra, Jasmyn Samaroo, Jason Wang, Jason Wolfe, Jay Chen, Jay Wang, Jean Harb, Jeffrey Han, Jeffrey Wang, Jennifer Zhao, Jeremy Chen, Jerene Yang, Jerry Tworek, Jesse Chand, Jessica Landon, Jessica Liang, Ji Lin, Jiancheng Liu, Jianfeng Wang, Jie Tang, Jihan Yin, Joanne Jang, Joel Morris, Joey Flynn, Johannes Ferstad, Johannes Heidecke, John Fishbein, John Hallman, Jonah Grant, Jonathan Chien, Jonathan Gordon, Jongsoo Park, Jordan Liss, Jos Kraaijeveld, Joseph Guay, Joseph Mo, Josh Lawson, Josh McGrath, Joshua Vendrow, Joy Jiao, Julian Lee, Julie Steele, Julie Wang, Junhua Mao, Kai Chen, Kai Hayashi, Kai Xiao, Kamyar Salahi, Kan Wu, Karan Sekhri, Karan Sharma, Karan Singhal, Karen Li, Kenny Nguyen, Keren Gu-Lemberg, Kevin King, Kevin Liu, Kevin Stone, Kevin Yu, Kristen Ying, Kristian Georgiev, Kristie Lim, Kushal Tirumala, Kyle Miller, Lama Ahmad, Larry Lv, Laura Clare, Laurance Fauconnet, Lauren Itow, Lauren Yang, Laurentia Romaniuk, Leah Anise, Lee Byron, Leher Pathak, Leon Maksin, Leyan Lo, Leyton Ho, Li Jing, Liang Wu, Liang Xiong, Lien Mamitsuka, Lin Yang, Lindsay McCallum, Lindsey Held, Liz Bourgeois, Logan Engstrom, Lorenz Kuhn, Louis Feuvrier, Lu Zhang, Lucas Switzer, Lukas Kondraciuk, Lukasz Kaiser, Manas Joglekar, Mandeep Singh, Mandip Shah, Manuka Stratta, Marcus Williams, Mark Chen, Mark Sun, Marselus Cayton, Martin Li, Marvin Zhang, Marwan Aljubeh, Matt Nichols, Matthew Haines, Max Schwarzer, Mayank Gupta, Meghan Shah, Melody Huang, Meng Dong, Mengqing Wang, Mia Glaese, Micah Carroll, Michael Lampe, Michael Malek, Michael Sharman, Michael Zhang, Michele Wang, Michelle Pokrass, Mihai Florian, Mikhail Pavlov, Miles Wang, Ming Chen, Mingxuan Wang, Minnia Feng, Mo Bavarian, Molly Lin, Moose Abdool, Mostafa Rohaninejad, Nacho Soto, Natalie Staudacher, Natan LaFontaine, Nathan Marwell, Nelson Liu, Nick Preston, Nick Turley, Nicklas Ansman, Nicole Blades, Nikil Pancha, Nikita Mikhaylin, Niko Felix, Nikunj Handa, Nishant Rai, Nitish Keskar, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Oona Gleeson, Pamela Mishkin, Patryk Lesiewicz, Paul Baltescu, Pavel Belov, Peter Zhokhov, Philip Pronin, Phillip Guo, Phoebe Thacker, Qi Liu, Qiming Yuan, Qinghua Liu, Rachel Dias, Rachel Puckett, Rahul Arora, Ravi Teja Mullapudi, Raz Gaon, Reah Miyara, Rennie Song, Rishabh Aggarwal, R. J. Marsan, Robel Yemiru, Robert Xiong, Rohan Kshirsagar, Rohan Nuttall, Roman Tsiupa, Ronen Eldan, Rose Wang, Roshan James, Roy Ziv, Rui Shu, Ruslan Nigmatullin, Saachi Jain, Saam Talaie, Sam Altman, Sam Arnesen, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Sarah Yoo, Savannah Heon, Scott Ethersmith, Sean Grove, Sean Taylor, Sebastien Bubeck, Sever Banesiu, Shaokyi Amdo, Shengjia Zhao, Sherwin Wu, Shibani Santurkar, Shiyu

Zhao, Shraman Ray Chaudhuri, Shreyas Krishnaswamy, Shuaiqi, Xia, Shuyang Cheng, Shyamal Anadkat, Simón Posada Fishman, Simon Tobin, Siyuan Fu, Somay Jain, Song Mei, Sonya Egoian, Spencer Kim, Spug Golden, S. Q. Mah, Steph Lin, Stephen Imm, Steve Sharpe, Steve Yadlowsky, Sulman Choudhry, Sungwon Eum, Suvansh Sanjeev, Tabarak Khan, Tal Stramer, Tao Wang, Tao Xin, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Degry, Thomas Shadwell, Tianfu Fu, Tianshi Gao, Timur Garipov, Tina Sriskandarajah, Toki Sherbakov, Tomer Kaftan, Tomo Hiratsuka, Tongzhou Wang, Tony Song, Tony Zhao, Troy Peterson, Val Kharitonov, Victoria Chernova, Vineet Kosaraju, Vishal Kuo, Vitchyr Pong, Vivek Verma, Vlad Petrov, Wanning Jiang, Weixing Zhang, Wenda Zhou, Wenlei Xie, Wenting Zhan, Wes McCabe, Will DePue, Will Ellsworth, Wulfie Bain, Wyatt Thompson, Xiangning Chen, Xiangyu Qi, Xin Xiang, Xinwei Shi, Yann Dubois, Yaodong Yu, Yara Khakbaz, Yifan Wu, Yilei Qian, Yin Tat Lee, Yinbo Chen, Yizhen Zhang, Yizhong Xiong, Yonglong Tian, Young Cha, Yu Bai, Yu Yang, Yuan Yuan, Yuanzhi Li, Yufeng Zhang, Yuguang Yang, Yujia Jin, Yun Jiang, Yunyun Wang, Yushi Wang, Yutian Liu, Zach Stubenvoll, Zehao Dou, Zheng Wu, and Zhigang Wang. OpenAI GPT-5 System Card, December 2025.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, Yichen Feng, Kelin Fu, Bofei Gao, Hongcheng Gao, Peizhong Gao, Tong Gao, Xinran Gu, Longyu Guan, Haiqing Guo, Jianhang Guo, Hao Hu, Xiaoru Hao, Tianhong He, Weiran He, Wenyang He, Chao Hong, Yangyang Hu, Zhenxing Hu, Weixiao Huang, Zhiqi Huang, Zihao Huang, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yongsheng Kang, Guokun Lai, Cheng Li, Fang Li, Haoyang Li, Ming Li, Wentao Li, Yanhao Li, Yiwei Li, Zhaowei Li, Zheming Li, Hongzhan Lin, Xiaohan Lin, Zongyu Lin, Chengyin Liu, Chenyu Liu, Hongzhang Liu, Jingyuan Liu, Junqi Liu, Liang Liu, Shaowei Liu, T. Y. Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yibo Liu, Yiping Liu, Yue Liu, Zhengying Liu, Enzhe Lu, Lijun Lu, Shengling Ma, Xinyu Ma, Yingwei Ma, Shaoguang Mao, Jie Mei, Xin Men, Yibo Miao, Siyuan Pan, Yebo Peng, Ruoyu Qin, Bowen Qu, Zeyu Shang, Lidong Shi, Shengyuan Shi, Feifan Song, Jianlin Su, Zhengyuan Su, Xinjie Sun, Flood Sung, Heyi Tang, Jiawen Tao, Qifeng Teng, Chensi Wang, Dinglu Wang, Feng Wang, Haiming Wang, Jianzhou Wang, Jiaxing Wang, Jinhong Wang, Shengjie Wang, Shuyi Wang, Yao Wang, Yejie Wang, Yiqin Wang, Yuxin Wang, Yuzhi Wang, Zhaoji Wang, Zhengtao Wang, Zhexu Wang, Chu Wei, Qianqian Wei, Wenhao Wu, Xingzhe Wu, Yuxin Wu, Chenjun Xiao, Xiaotong Xie, Weimin Xiong, Boyu Xu, Jing Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ziyao Xu, Junjie Yan, Yuzi Yan, Xiaofei Yang, Ying Yang, Zhen Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Xingcheng Yao, Wenjie Ye, Zhuorui Ye, Bohong Yin, Longhui Yu, Enming Yuan, Hongbang Yuan, Mengjie Yuan, Haobing Zhan, Dehao Zhang, Hao Zhang, Wanlu Zhang, Xiaobin Zhang, Yangkun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Haotian Zhao, Yikai Zhao, Huabin Zheng, Shaojie Zheng, Jianren Zhou, Xinyu Zhou, Zaida Zhou, Zhen Zhu, Weiyu Zhuang, and Xinxing Zu. Kimi K2: Open Agentic Intelligence, July 2025.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop Questions via Single-hop Question Composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475.

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, Zhongwang Zhang, Xinmiao Yu, Dingchu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, Minhao Cheng, Shuai Wang, Hong Cheng, and Jingren Zhou. ReSum: Unlocking Long-Horizon Search Intelligence via Context Summarization, March 2026.

Xingrun Xing, Zhiyuan Fan, Jie Lou, Guoqi Li, Jiajun Zhang, and Debing Zhang. PretrainZero: Reinforcement Active Pretraining, December 2025.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic Memory for LLM Agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, October 2025.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schütze, Volker Tresp, and Yunpu Ma. Memory-

R1: Enhancing Large Language Model Agents to Manage and Utilize Memories via Reinforcement Learning. https://arxiv.org/abs/2508.19828v5, August 2025.

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 Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models, March 2023.

Rui Ye, Zhongwang Zhang, Kuan Li, Huifeng Yin, Zhengwei Tao, Yida Zhao, Liangcai Su, Liwen Zhang, Zile Qiao, Xinyu Wang, Pengjun Xie, Fei Huang, Siheng Chen, Jingren Zhou, and Yong Jiang. AgentFold: Long-Horizon Web Agents with Proactive Context Management, October 2025.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. CompAct: Compressing Retrieved Documents Actively for Question Answering. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21424–21439, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1194.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. MemAgent: Reshaping Long-Context LLM with Multi-Conv RL-based Memory Agent. In *The Fourteenth International Conference on Learning Representations*, October 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An Open-Source LLM Reinforcement Learning System at Scale, May 2025.

Qianhao Yuan, Jie Lou, Zichao Li, Jiawei Chen, Yaojie Lu, Hongyu Lin, Le Sun, Debing Zhang, and Xianpei Han. MemSearcher: Training LLMs to Reason, Search and Manage Memory via End-to-End Reinforcement Learning, May 2026.

Eric Zelikman, Georges Raif Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah Goodman. Quiet-STaR: Language Models Can Teach Themselves to Think Before Speaking. In *First Conference on Language Modeling*, August 2024.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Bryan Kian Hsiang Low, and Paul Pu Liang. MEM1: Learning to Synergize Memory and Reasoning for Efficient Long-Horizon Agents. In *The Fourteenth International Conference on Learning Representations*, October 2026.

A Prompt Template

MemTrain employs three prompt templates, as illustrated below. For the end-to-end masked reconstruction task, we adopt the prompt design from MemAgent (Yu et al., 2026) and set the problem as a fixed masked prediction instruction. Specifically, the memory generation prompt is applied iteratively until all text chunks have been processed, after which the answer generation prompt is used to produce the final output. For the intermediate memory recall task, we introduce the placeholder [TARGET] to distinguish it from [MASK], thereby preventing the LLM from being confused about which reconstruction objective to perform.

End-to-End Memory Generation Prompt

```
You are presented with a problem, a section of an article that may
contain the answer to the problem, and a previous memory. Please
read the provided section carefully and update the memory with the
new information that helps to answer the problem. Be sure to retain
 all relevant details from the previous memory while adding any new
, useful information.
<problem> Based on the entire article, which entity is represented
by the [MASK] placeholder? </problem>
<memory> {memory} </memory>
<section> {chunk} </section>

Updated memory:
```

End-to-End Answer Generation Prompt

```
You are presented with a problem and a previous memory. Please
answer the problem based on the previous memory and put the answer
in boxed{}.
<problem> Based on the entire article, which entity is represented
by the [MASK] placeholder? </problem>
<memory> {memory} </memory>
```

Intermediate Memory Recall Prompt

```
You are presented with a memory from previously read sections, and
a section containing placeholder [TARGET]. Based on the memory,
please predict the entity marked by [TARGET] and put the answer in
boxed{}.
<memory> {memory} </memory>
<section> {masked chunk} </section>
```

B MemTrain Algorithm

Algorithm 1 describes data construction, rollout collection, reward coupling, the conversion of grouped trajectories into turn-level training records, and the update of the policy π_θ. The data-construction stage implements Step (1) of Figure 2, following Section 3.2. Following Figure 2, a document contains $T-1$ chunks and each E2E trajectory contains T turns, including a final answer turn with no new chunk. The instruction q^E uses the appropriate memory-generation or answer-generation template; q^I is the IMR template. The operator $\oplus$ concatenates prompt components. The initial parameters are θ_{ref}, and $\pi_{\text{old}} = \pi_{\theta_{\text{old}}}$; the reference policy remains fixed throughout training. Here, $\mathrm{Retrieve}$ returns semantically related passages, Sample draws random passages, and $+$ concatenates passage lists. $\mathrm{MaskAll}$ replaces every occurrence of y in the concatenated document; Split divides it into chunks of length L, with $T-1$ denoting the resulting number of chunks. $\mathrm{RandomEntity}$ selects a visible, unmasked entity, and $\mathrm{MaskEntity}$ masks the selected entity for IMR. Implementation-level dynamic sampling is omitted to focus on the core data flow.

From grouped rewards to training records. For each document, the G_1 E2E trajectories form one advantage group. Each trajectory also produces its own IMR prompt, whose G_2 responses form a separate advantage group. Thus, IMR responses associated with different memories do not share a reward baseline. The E2E advantage $\hat{A}_i^E$ is assigned to all T turns of trajectory i, whereas $\hat{A}_{i,j}^I$ is assigned only to IMR response (i,j). Both are mean-centered without standard-

Algorithm 1 MemTrain: data construction, rollouts, and policy optimization

Input: Wikipedia corpus $\mathcal{W}$; initial policy π_{ref}; dataset size M; passage counts N, n_1; chunk length L; rollout group sizes G_1, G_2; coefficient λ; steps S
Output: Trained policy π_θ

$\theta \leftarrow \theta_{\text{ref}}$; $\mathcal{D} \leftarrow \varnothing$
for $d = 1, \dots, M$ **do**
  Pivot passage $p \sim \mathcal{W}$
  $P_{\text{rel}} \leftarrow \text{Retrieve}(\mathcal{W}, p, n_1)$
  $P_{\text{rand}} \leftarrow \text{Sample}(\mathcal{W}, N - n_1 - 1)$
  $x \leftarrow \text{Concat}(\text{Shuffle}([p] + P_{\text{rel}} + P_{\text{rand}}))$
  $y \leftarrow \text{RandomEntity}(p)$
  $\tilde{x} \leftarrow \text{MaskAll}(x, y, \texttt{[MASK]})$
  $c_{1:T-1} \leftarrow \text{Split}(\tilde{x}, L)$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(c_{1:T-1}, y)\}$
end for
for $s = 1, \dots, S$ **do**
  $\theta_{\text{old}} \leftarrow \theta$; $\mathcal{B} \leftarrow \varnothing$
  $\mathcal{M} \leftarrow \text{SampleBatch}(\mathcal{D})$
  for each $(c_{1:T-1}, y) \in \mathcal{M}$ **do**
    for $i = 1, \dots, G_1$ **do** ▷ E2E Rollout Group
      $o^E_{i,0} \leftarrow \varnothing$
      for $t = 1, \dots, T-1$ **do** ▷ Memory Maintenance
        $x^E_{i,t} \leftarrow q^E \oplus o^E_{i,t-1} \oplus c_t$
        $o^E_{i,t} \sim \pi_{\text{old}}(\cdot \mid x^E_{i,t})$
      end for
      $x^E_{i,T} \leftarrow q^E \oplus o^E_{i,T-1}$ ▷ Final Answer Prediction
      $o^E_{i,T} \sim \pi_{\text{old}}(\cdot \mid x^E_{i,T})$
      $\hat{y}^E_i \leftarrow \text{ExtractAnswer}(o^E_{i,T})$
      Sample $k \in \{2, \dots, T-1\}$ and $l \in \{1, \dots, k-1\}$
      $\tilde{y}_i \leftarrow \text{RandomEntity}(c_l)$
      $\tilde{c}_l \leftarrow \text{MaskEntity}(c_l, \tilde{y}_i, \texttt{[TARGET]})$
      $x^I_i \leftarrow q^I \oplus o^E_{i,k} \oplus \tilde{c}_l$
      for $j = 1, \dots, G_2$ **do** ▷ IMR Rollout Group
        $o^I_{i,j} \sim \pi_{\text{old}}(\cdot \mid x^I_i)$
        $\hat{y}^I_{i,j} \leftarrow \text{ExtractAnswer}(o^I_{i,j})$
        $R^I_{i,j} \leftarrow \mathbb{I}[\hat{y}^I_{i,j} = \tilde{y}_i]$
      end for
      $R^E_i \leftarrow \mathbb{I}[\hat{y}^E_i = y] + \frac{\lambda}{G_2} \sum_{j=1}^{G_2} R^I_{i,j}$ ▷ Coupled reward for E2E
    end for
    for $i = 1, \dots, G_1$ **do**
      $\hat{A}^E_i \leftarrow R^E_i - \frac{1}{G_1} \sum_{i=1}^{G_1} R^E_i$
      for $t = 1, \dots, T$ **do** ▷ Broadcast $\hat{A}^E_i$ to all turns
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{\text{PackTurn}(x^E_{i,t}, o^E_{i,t}, \hat{A}^E_i, \pi_{\text{old}})\}$
      end for
      for $j = 1, \dots, G_2$ **do**
        $\hat{A}^I_{i,j} \leftarrow R^I_{i,j} - \frac{1}{G_2} \sum_{j=1}^{G_2} R^I_{i,j}$ ▷ Only for response (i, j)
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{\text{PackTurn}(x^I_i, o^I_{i,j}, \hat{A}^I_{i,j}, \pi_{\text{old}})\}$
      end for
    end for
  end for
  $\theta \leftarrow \text{GRPOUpdate}(\theta, \mathcal{B})$ ▷ Eq. (4)
end for
return π_θ

deviation normalization. $\text{PackTurn}(x, o, A, \pi_{\text{old}})$ stores the prompt x, response o, old-policy token probabilities, and advantage A broadcast to the response tokens. Only generated response tokens receive the advantage and contribute to the policy loss. The prompt includes the sampled memory and, when applicable, the current or masked historical chunk. Each document contributes $G_1 T + G_1 G_2$ records before filtering. GRPOUpdate updates the policy using the records in $\mathcal{B}$ to maximize the objective in Eq. (4).