$(\mathcal{LC})^2$: LEARNING CONSTANT-SIZE LATENT CONTEXT FOR LANGUAGE MODEL AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Language model agents accumulate increasingly long contexts of observations, actions, tool outputs, and intermediate reasoning. Without proper context management, as the context grows, agents' performance can degrade significantly. Existing context management methods primarily shorten agent context through textual summarization, token selection, retrieval, or via external symbolic memory. Other work has also explored latent tokens to represent conversation context, but it remains unclear whether latent tokens can represent agent contexts and preserve the information useful for determining future agent actions. In this work, we propose $(\mathcal{LC})^2$: a framework for **l**earning a **c**onstant-size **l**atent **c**ontext for a *frozen* language model agent. We leverage the strong representational power of the continuous latent token space: *even as the rounds of agent interaction increases, the number of (latent) tokens representing the context remains constant*. Concretely, we train a lightweight compressor to compress past context history into a constant number of latent tokens that we insert directly into the frozen language model's embedding sequence. We use a two-stage training procedure: behavioral supervised learning first teaches the compressor to preserve the full context, followed by reinforcement learning that directly optimizes the compressor using the downstream task reward. We argue that this training procedure can learn a compact latent representation of an agent's context, rather than relying on pure natural-language text summarization. Results on three widely used agentic benchmarks show that this method achieves similar performance while dramatically reducing token cost.

1 INTRODUCTION

Recently, language model agents (Su et al., 2024; Schick et al., 2023; Yao et al., 2023; Schick et al., 2023), built on large language models (LLMs), have become increasingly capable in many areas such as web navigation (Deng et al., 2023; Zhou et al., 2024), information seeking (Nakano et al., 2021; Yao et al., 2023), and deep research (Zheng et al., 2025; Li et al., 2025). The most fundamental and widely used agentic paradigm is ReAct (Yao et al., 2023), where agents interact with the external environment, invoke tools, observe outputs, and reason about actions turn by turn. Unlike conventional single-turn language modeling, an agent's decision at each turn is conditioned on an evolving *context* consisting of previous turns' observations, actions, tool outputs, and intermediate reasoning.

As a result, context grows continuously with the number of turns, causing several issues. *First*, the growing context significantly increases the inference cost. *Second*, the growing context may exceed the language model's context window, causing the agent to lose information from previous turns. *Third*, even if the context stays within the language model's context window, research shows that the effective context window is much smaller than the physical context window[1], a phenomenon known as *context rot* (Hong et al., 2025; Liu et al., 2024). Context rot will significantly degrade agents' performance. As such, effective context management for language model agents has become an important research question. The challenge is to manage the context so that the agent can make the right decisions.

[1]By physical context window, we mean the context window length determined by the model architecture.

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.

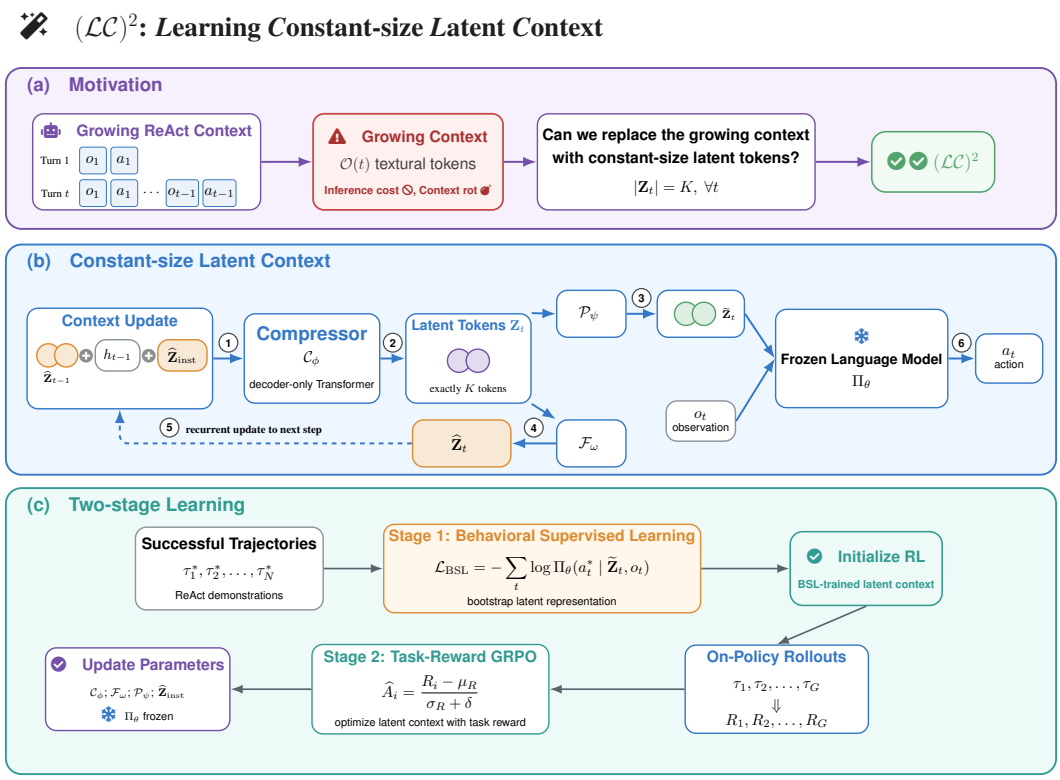


Figure 1: **Overview of** $(\mathcal{LC})^2$. (a) ReAct-style language-model agents accumulate an increasingly long context, driving up inference costs and causing context rot. (b) $(\mathcal{LC})^2$ replaces the growing context with constant-size latent tokens. ① The compressor $\mathcal{C}_\phi$ integrates the previous latent tokens $\widehat{\mathbf{Z}}_{t-1}$, the latest interaction h_{t-1}, and latent instruction tokens $\widehat{\mathbf{Z}}_{\text{inst}}$ to ② output $\mathbf{Z}_t$. ③ $\mathcal{P}_\psi$ maps $\mathbf{Z}_t$ into the frozen language model's embedding space, ④ while $\mathcal{F}_\omega$ maps $\mathbf{Z}_t$ into the compressor embedding space ⑤ for the next turn. ⑥ Finally, a_t is sampled from the frozen language model Π_θ conditioned on $\widetilde{\mathbf{Z}}_t$ and o_t. Orange represents the compressor input embedding space, purple represents the compressor transformer's last layer hidden space, and green represents the language model input embedding space. (c) $(\mathcal{LC})^2$ is first trained with behavioral supervised learning on successful trajectories and subsequently optimized with trajectory-level task rewards using GRPO, while the language-model policy remains frozen.

Existing approaches (Wu et al., 2025; Ye et al., 2025; Sun et al., 2025) for agent context management, however, operate predominantly in the **natural-language space**: they select, discard, rewrite, or retrieve past interactions into shorter textual contexts, based on either heuristics, optimization algorithms, or other language model agents, which could introduce additional token costs. In contrast, another line of research has explored compressing long contexts into **latent token representations**. Gisting (Mu et al., 2023), AutoCompressors (Chevalier et al., 2023), and ICAE (Ge et al., 2024), for example, represent long textual inputs using learned latent tokens or vectors. While these studies show the potential of latent tokens for context representations, they focus on prompts, documents, or conventional language-model contexts rather than the agent setting. This raises a natural research question: ***Can an agent represent its growing context history as latent tokens?***

Representing the context in latent token space is more appealing because, in theory, the same number of continuous latent tokens can store much more information than discrete tokens. However, directly representing agent context in latent token space is not without its challenges: *First*, unlike single-turn context representation, the agent context is *dynamic* and *stateful*; each turn's new context (including observations, actions, reasoning, etc) is conditioned on the previous context representation. *Second*, single-turn context representation focuses on preserving the semantic information of the context, while a good agent context representation should preserve the information required to perform future

agent actions. These fundamentally different goals may lead to different decisions about which information to select and which to discard.

In this work, we introduce $(\mathcal{LC})^2$: a framework that **L**earns **C**onstant-size **L**atent **C**ontext for language model agents. See Figure 1 for an overview. We train a separate lightweight model, a compressor, to compress the agent's accumulated context into a constant number K of latent tokens, while keeping the original language model frozen. We insert the latent tokens directly into the frozen language model's embedding sequence. Crucially, K is **independent of the number of turns**: while the raw interaction context continually grows, the latent context representation remains constant. Furthermore, we **decouple context compression from language model policy execution**. This modular design substantially reduces training cost, preserves the underlying language model's capabilities, and lets us optimize the latent representation specifically for downstream agent behavior. We train the compressor similarly to how language models are post-trained these days (DeepSeek-AI, 2025; Team, 2025; 2026): behavior supervised learning followed by reinforcement learning. We show that this two-stage training recipe can also help train the compressor, albeit with different training objectives and for completely different purposes.

We evaluate $(\mathcal{LC})^2$on three representative agent benchmarks: ALFWorld (Shridhar et al., 2021), Webshop (Yao et al., 2022), and HotpotQA (Yang et al., 2018). Our method achieves on-par or better performance than many context management methods while significantly reducing token cost. Further analyses show that our method is also effective for long-horizon settings, that each training step matters, and that the constant K balances representation capacity and generalization.

2 Related Work

2.1 Agent Context Management

Language model agents accumulate observations, actions, tool outputs, and intermediate reasoning into the context, which can readily exceed the context limit of the agent. As such, context management becomes essential for agent performance. Some common approaches manage the context in *natural-language space* through heuristic selection, summarization, or hierarchical organization. HiAgent (Hu et al., 2025) organizes the context into hierarchical working memory. It prompts LLMs to formulate and replace subgoals with summarized observations. ACON (Kang et al., 2025) optimizes compression strategies by attending to failures and distills the updated strategy into smaller compressors. Rather than explicitly summarizing the entire context, PACE (Wei et al., 2026) dynamically adjusts context information granularity based on its predicted relevance. These approaches show that heuristic-based retention and summarization can substantially reduce the agent's context. However, they may increase the token cost if the context or heuristics themselves come from another language model.

Another line of work introduces a *separate learned component* for context management. CoMem (Zhang et al., 2026a) decouples context management from language model policy execution through an auxiliary long-context model, which adds cost. AgentFold (Ye et al., 2025) proactively consolidates interaction histories at different granularities, allowing long-horizon web agents to discard intermediate details while preserving information needed for future steps. These approaches motivate separating *context management* from *language model policy execution*, but still mainly operate in natural language space.

More recent work directly *learns* context-management. MEM1 (Zhou et al., 2025) jointly learns reasoning and memory consolidation through reinforcement learning, but it requires fine-tuning the language model itself. SUPO (Lu et al., 2025) integrates context summarization into end-to-end agentic RL, jointly optimizing reasoning and context summarization. Similarly, Memory-as-Action (Zhang et al., 2026b) integrates memory-editing operations into the agent action space and optimizes the language model via end-to-end agentic RL. Context-Folding (Sun et al., 2025) designs a reinforcement learning algorithm to encourage task decomposition and context management. These works shift from *heuristically designed* context management to end-to-end learning of context representations.

Despite different mechanisms, these approaches predominantly expose the compressed context to the language model through *discrete natural-language tokens*. Our work investigates a complemen-

tary representation choice: we compress the evolving interaction context into *continuous constant-size latent tokens*. By training a separate compressor and keeping the language model frozen, our approach retains the modular separation between context compression and language model policy execution. However, we change the representation into a more compact continuous latent token space.

2.2 Latent Representations for Context Compression

Another line of research represents long textual contexts with compact, learned latent representations. Gist Tokens (Mu et al., 2023) compress prompts into a small set of learned tokens with modified attention masks. It shows that prompt instructions can be preserved in a "gist" token space. ICAE (Ge et al., 2024) extends this idea to long contexts by compressing text into compact memory slots that a language model can condition on directly. Activation Beacon (Zhang et al., 2025) compresses LLM activations through trainable beacon tokens and enables recurrent processing of contexts beyond the model's context window. xRAG (Cheng et al., 2024) trains a modality bridge, which maps dense retrieval embeddings directly into the LLM representation space. It represents documents with few tokens while keeping both the retriever and the language model frozen.

These works show the potential of compact latent representations for long contexts, but primarily focus on *static and stateless prompts and documents*. Agent context, by contrast, is *dynamic and stateful*. Our work bridges these two directions by learning a **constant-size latent representation of an evolving agent context** with a separate compressor, while keeping the language model frozen.

3 Method

We propose $(\mathcal{LC})^2$: a framework for **l**earning a **c**onstant-size **l**atent **c**ontext representation for language model agents. We do not concatenate an ever-growing context for each turn. Instead, we train a lightweight compressor $\mathcal{C}_\phi$ to integrate the current turn's new information into a constant number of latent tokens. The compressor itself is a small pre-trained decoder-only transformer.

To achieve this, we introduce K learnable latent tokens $\widehat{\mathbf{Z}}_{\text{inst}}$ at the compressor input embedding space [2]. Intuitively, $\widehat{\mathbf{Z}}_{\text{inst}}$ represents the instruction to concatenate the previous turn's information with the current turn. Moreover, we can directly read out $\mathbf{Z}$ from the last K positions of the last layer's hidden state of the compressor transformer.

The compressor output hidden state is not aligned with either its input embedding space or the language model's input embedding space. To deal with this, we introduce two learned non-linear maps. $\mathcal{F}_\omega$ maps the output hidden state space back into the compressor input embedding space, while $\mathcal{P}_\psi$ maps the compressor output hidden state space into the language model's embedding space. The language model Π_θ remains frozen during training. We train the compressor and the two maps using behavior supervised learning followed by reinforcement learning with Group Relative Policy Optimization (GRPO; Shao et al., 2024), with downstream task rewards.

3.1 Problem Formulation

We consider the language model agent, backed by the language model policy Π_θ, interacting with an environment ξ. At step t, the agent receives an observation o_t from the environment and generates an action a_t, which may include intermediate reasoning and/or tool calls. We denote each turn's information at step t by h_t, where $h_t = (o_t, a_t)$. We define the raw accumulated context before step t as

$$\mathcal{H}_{<t} = (h_1, \ldots, h_{t-1}).$$

In a ReAct-style agent, we have:

$$a_t \sim \Pi_\theta\left(\cdot \mid \mathcal{H}_{<t}, o_t\right)$$

[2] We use $\mathbf{Z}$ to represent latent tokens at the compressor transformer's last layer hidden state space, $\widehat{\mathbf{Z}}$ to represent latent tokens at the compressor input embedding space, and $\widetilde{\mathbf{Z}}$ to represent latent tokens at the language model input embedding space.

3.2 Method Overview and Inference

In this subsection, we describe the $(\mathcal{LC})^2$inference procedure and provide an overview of our method. We use d_c for the compressor transformer's model dimension and d as the language model's dimension. At step t, we use a compressor to output K latent tokens $\mathbf{Z}_t$. Concretely,

$$\mathbf{Z}_t = \mathcal{C}_\phi\left(\widehat{\mathbf{Z}}_{t-1}, h_{t-1}, \widehat{\mathbf{Z}}_{\text{inst}}\right) \in \mathbb{R}^{K\times d_c} \tag{1}$$

$\widehat{\mathbf{Z}}_{\text{inst}} \in \mathbb{R}^{K\times d_c}$ are learnable K latent tokens that intuitively represent the instruction to merge previous steps' information and the current step's observation. $\mathbf{Z}_t$ is taken from the last K positions of the last layer's hidden state of the compressor transformer. The compressor transformer's last-layer hidden space is not aligned with the compressor transformer's input embedding space or the language model's input embedding space. We therefore introduce two learnable non-linear maps.

1. $\mathcal{F}_\omega : \mathbb{R}^{d_c} \to \mathbb{R}^{d_c}$ maps the compressor's last layer hidden state space back to its input embedding space.
$$\widehat{\mathbf{Z}}_t = \mathcal{F}_\omega\left(\mathbf{Z}_t\right) \in \mathbb{R}^{K\times d_c} \tag{2}$$
2. $\mathcal{P}_\psi : \mathbb{R}^{d_c} \to \mathbb{R}^{d}$ maps the compressor's last layer hidden state space to the language model's input embedding space.
$$\widetilde{\mathbf{Z}}_t = \mathcal{P}_\psi\left(\mathbf{Z}_t\right) \in \mathbb{R}^{K\times d} \tag{3}$$

a_t is then sampled as:

$$a_t \sim \Pi_\theta\left(\cdot \mid \widetilde{\mathbf{Z}}_t, o_t\right) \tag{4}$$

In summary, at step t, the inference procedure is as follows:

$$\boxed{\begin{aligned}
\mathbf{Z}_t &= \mathcal{C}_\phi\left(\widehat{\mathbf{Z}}_{t-1}, h_{t-1}, \widehat{\mathbf{Z}}_{\text{inst}}\right)\\
\widehat{\mathbf{Z}}_t &= \mathcal{F}_\omega\left(\mathbf{Z}_t\right)\\
\widetilde{\mathbf{Z}}_t &= \mathcal{P}_\psi\left(\mathbf{Z}_t\right)\\
a_t &\sim \Pi_\theta\left(\cdot \mid \widetilde{\mathbf{Z}}_t, o_t\right)
\end{aligned}} \tag{5}$$

Non-linear map details. $\mathcal{F}_\omega$ and $\mathcal{P}_\psi$ are learnable shallow nonlinear token-wise mappings.

Each map has the following format:

$$M(\mathbf{x}) = W_2\sigma\left(W_1\,\mathrm{LN}(\mathbf{x}) + b_1\right) + b_2,$$

where LN is layer normalization (Ba et al., 2016) and σ is a non-linear activation function; we choose GELU (Hendrycks & Gimpel, 2016). The two maps have independent parameters.

Algorithm 1 summarizes the inference procedure.

3.3 Behavior Supervised Learning

Directly training the compressor using reinforcement learning is challenging and can cause training instability. As such, we bootstrap by first using behavior supervised learning. In this stage, we train the compressor to mimic the behavior of a ReAct-style trajectory. Concretely,

Let

$$\tau^* = (o_1, a_1^*, \ldots, o_T, a_T^*)$$

be a successful trajectory from dataset $\mathcal{D}$. During training, at each step t, we teacher-force the demonstrated action a_{t-1}^*. We want the frozen language model to reproduce the demonstrated action; thus, we design the loss function for the behavior supervised learning as follows:

$$\mathcal{L}_{\text{BSL}} = -\mathbb{E}_{\tau^*\sim\mathcal{D}}\left[\sum_{t=1}^{T}\log\Pi_\theta\left(a_t^* \mid \mathcal{P}_\psi(\mathbf{Z}_t), o_t\right)\right]. \tag{6}$$

For an autoregressive action $a_t^* = (y_{t,1}^*, \ldots, y_{t,L_t}^*)$, where y is a single token and L_t is the token length of a_t, the likelihood can be further decomposed as:

$$\log \Pi_\theta \left(a_t^* \mid \mathcal{P}_\psi(\mathbf{Z}_t), o_t\right) = \sum_{l=1}^{L_t} \log \Pi_\theta \left(y_{t,l}^* \mid \mathcal{P}_\psi(\mathbf{Z}_t), o_t, y_{t,<l}^*\right). \tag{7}$$

The language model parameters θ remain frozen. Gradients propagate through the language model into the compressor and the non-linear maps. The trainable parameters are therefore

$$\Theta_{\mathrm{BSL}} = \{\phi, \omega, \psi, \widehat{\mathbf{Z}}_{\mathrm{inst}}\}. \tag{8}$$

We optimize

$$\Theta_{\mathrm{BSL}} \leftarrow \Theta_{\mathrm{BSL}} - \eta \nabla_{\Theta_{\mathrm{BSL}}} \mathcal{L}_{\mathrm{BSL}}. \tag{9}$$

Algorithm 2 summarizes this training stage.

3.4 Reinforcement Learning with GRPO

After behavior supervised learning, we further train the compressor directly with GRPO (Shao et al., 2024) using trajectory-level task rewards. The trainable parameters are the same and are initialized from behavior supervised learning (i.e., $\Theta_{\mathrm{BSL}}^* = \Theta_{\mathrm{GRPO}}^{(0)}$).

For each task instance x, we sample a group of G trajectories $\{\tau_i\}_{i=1}^G$ using the current compressor and frozen language model. Each trajectory is independent of the others. Upon termination, trajectory τ_i receives task reward R_i. We compute its group-relative advantage as

$$\widehat{A}_i = \frac{R_i - \mu_R}{\sigma_R + \delta}, \qquad \mu_R = \frac{1}{G}\sum_{j=1}^{G} R_j, \tag{10}$$

where σ_R is the standard deviation of rewards within the group and $\delta > 0$ is a small constant for numerical stability.

For token $y_{i,t,l}$ [3], define

$$p_{\Theta_{\mathrm{GRPO}}}(y_{i,t,l}) = \Pi_\theta\Big(y_{i,t,l} \mid \mathcal{P}_\psi(\mathbf{Z}_{i,t}^{\Theta_{\mathrm{GRPO}}}), o_{i,t}, y_{i,t,<l}\Big), \tag{11}$$

where $\mathbf{Z}_{i,t}^{\Theta_{\mathrm{GRPO}}}$ is obtained by recurrently applying Eq. 5.

Let $\Theta_{\mathrm{GRPO}}^{\mathrm{old}}$ denote the parameters used to generate the rollout. The token-level importance ratio is

$$r_{i,t,l}(\Theta_{\mathrm{GRPO}}) = \frac{p_{\Theta_{\mathrm{GRPO}}}(y_{i,t,l})}{p_{\Theta_{\mathrm{GRPO}}^{\mathrm{old}}}(y_{i,t,l})}. \tag{12}$$

We then optimize against the clipped GRPO objective

$$\begin{aligned}\mathcal{J}_{\mathrm{GRPO}}(\Theta_{\mathrm{GRPO}}) = \mathbb{E}_{i,t,l}\Big[\min\big(&r_{i,t,l}(\Theta_{\mathrm{GRPO}})\hat{A}_i, \\ &\mathrm{clip}(r_{i,t,l}(\Theta_{\mathrm{GRPO}}), 1-\varepsilon, 1+\varepsilon)\hat{A}_i\big)\Big],\end{aligned} \tag{13}$$

where ε controls the clipping range. For simplicity, we ignore the KL regularization term.

Truncated gradient propagation. The recurrent nature of $\mathbf{Z}_t$ creates a gradient dependency chain, like a recurrent update in an RNN. In practice, especially in long-horizon interactions, we observe some training instability due to this effect. To remedy this, we use Truncated Backpropagation Through Time (TBPTT; Williams & Peng, 1990), where the gradient dependency can span at most M_{TBPTT} steps. In mathematical terms,

$$\mathbf{Z}_t \to \mathbf{Z}_{t+1} \to \cdots \to \mathbf{Z}_{t+M_{\mathrm{TBPTT}}}, \qquad \mathrm{stopgrad}(\mathbf{Z}_{t+M_{\mathrm{TBPTT}}}). \tag{14}$$

The truncation affects only gradient propagation. The forward computation does not change. Algorithm 3 summarizes the RL stage.

[3] lth token at step t in trajectory i

4 EXPERIMENTS

We evaluate our method on three representative agentic benchmarks: **ALFWorld** (Shridhar et al., 2021), **WebShop** (Yao et al., 2022), and **HotpotQA** (Yang et al., 2018), representing an embodied intelligence environment, a web shopping environment, and a search-based QA environment. Our evaluation section aims to answer the following research questions: (1) How well does our method perform, both in terms of performance and token counts? (2) How does the number of latent tokens K affect in-distribution and out-of-distribution performance? (3) How well does our method perform under long-horizon scenarios? (4) How much does each training stage contribute to the performance?

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate on **ALFWorld**, **WebShop**, and **HotpotQA**. **ALFWorld** is a text-based interactive embodied environment with six task types and two evaluation settings: seen tasks with in-distribution environments and unseen tasks with out-of-distribution environments. **WebShop** is a text-based simulated web shopping environment, and the agent's task is to navigate the webpages to achieve the goals in the instruction. **HotpotQA** is a multi-step question-answering benchmark where the agent can use search tools to answer questions.

Metrics. For **ALFWorld**, we use success rate as the performance metric, where 1 represents the agent successfully completes the task. For **WebShop**, we use the reward score as the performance metric. The agent can get partial scores even if it does not complete the task perfectly. For **HotpotQA**, we report both the Exact Match (EM) and F1 scores. EM score reports whether the agent's answer exactly matches the golden answer after string normalization, while the F1 score measures token-level overlap between the agent's answer and the golden answer. All reported numbers are re-scaled to 1-100 for better readability.

Baselines. We compare our method against three representative context management methods. (1) **ReAct** (Yao et al., 2023), where at each turn, the context simply concatenates the previous turn's history. (2) **Sliding window** is a widely used context management method in practice, where the method maintains a fixed context budget. If the context exceeds the budget, it discards older history. (3) **Summrization** (Wu et al., 2025), which periodically summarizes the context by calling another agent. We report additional details and parameters about our method and the baseline in Appendix B.

4.2 MAIN RESULTS

Table 1: Main results on **ALFWorld**, **WebShop**, and **HotpotQA**. We report success rates on the seen and unseen splits of **ALFWorld**, task score on **WebShop**, and exact match (EM) and F1 on **HotpotQA**. Tokens denote the average number of input tokens for the evaluation. Higher task performance and lower token consumption are better. We **highlight** the best result and underline the second-best result.

	ALFWorld			WebShop		HotpotQA		
Method	Seen ↑	Unseen ↑	Tokens ↓	Score ↑	Tokens ↓	EM ↑	F1 ↑	Tokens ↓
ReAct	59.5	52.1	23.1M	46.2	9.5M	<u>34.1</u>	<u>37.7</u>	3.3M
Sliding Window	43.1	39.9	6.7M	33.1	2.1M	29.1	33.4	1.9M
Summarization	53.4	50.0	3.2M	41.7	1.9M	32.1	35.4	2.0M
Ours ($K = 16$)	<u>64.5</u>	**62.1**	**1.5M**	<u>50.9</u>	**0.8M**	33.0	35.7	**0.3M**
Ours ($K = 32$)	**67.1**	<u>60.4</u>	<u>1.6M</u>	**51.6**	<u>0.8M</u>	**35.7**	**39.6**	<u>0.4M</u>

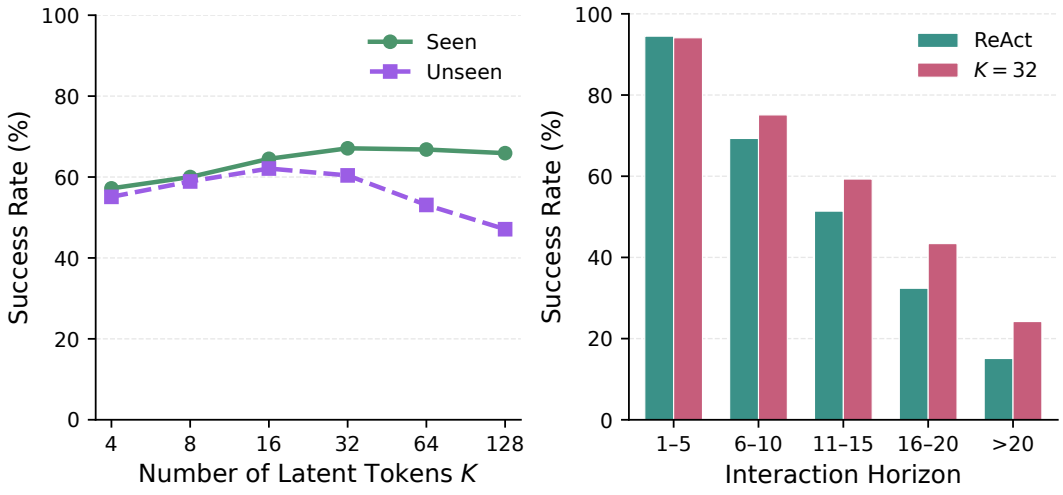
Table 1 presents our main results of three benchmarks. We report both the total number of tokens and the task performance for each method.

In terms of task performance, our methods consistently outperform the ReAct baseline across all three benchmarks. In particular, in the **ALFWorld** seen setting, our best method $K = 32$ outper-

forms ReAct by 7.6%; in unseen settings, our best method $K = 16$ outperforms ReAct by 10.0%. The sliding window method performs poorly on **ALFWorld**. We conjecture that **ALFWorld** has the longest horizon over these three benchmarks, and the sliding window method may miss important context from early histories. Interestingly, we achieve **ALFWorld** seen best result when $K = 32$, while we achieve **ALFWorld** unseen result when $K = 16$. This suggests that although $K = 32$ gives us the best in-distribution performance, $K = 16$ gives us the best out-of-distribution performance. Choosing K is therefore a trade-off between capability and generalization.

In terms of tokens, our method uses the fewest tokens by design. On average, ReAct uses 10x as many tokens as our method. The sliding window and summarization method also uses far fewer tokens than ReAct. The summarization method generally uses fewer tokens than the sliding window method, except in the **HotpotQA** benchmark. Note that $K = 32$ uses only slightly more tokens than $K = 16$. This suggests that increasing K only moderately increases the token count, and most tokens in our method come from each turn's new information, which is unavoidable.

In summary, our method achieves the best overall performance among all baselines while significantly reducing the total number of tokens.



(a) Effect of K.

(b) Performance across different interaction horizons.

Figure 2: (a) Success rate of seen and unseen settings in **ALFWorld** across different K. (b) Success rate of $K = 16$ and $K = 32$ across different interaction horizons in **ALFWorld**, aggregated between seen and unseen.

4.3 Effect of K: Number of Latent Tokens

In Table 1, we observe that larger K does not necessarily lead to better unseen performance in **ALFWorld**. We conjecture that large K offers more powerful representation capabilities but may also overfit, leading to worse performance on out-of-distribution unseen tasks. To analyze this further and quantify the trade-off between capacity and generalization, we use K values from 4 to 128 and compare seen and unseen performance.

As shown in Figure 2a, for seen performance, $K = 32$ yields the best among all values. Smaller K may lack the representational power to learn the training data, while larger K may overfit to the training data. In practice, the performance of larger K plateaus in seen settings. However, in unseen settings, the story is different. Optimal K shifts from 32 to 16. Although smaller K generally remains suboptimal, larger K degrades significantly. This shows that larger K overfits significantly to the seen training data and generalizes poorly beyond the training distribution.

Overall, this study shows that the choice of K has profound implications: smaller K may lack the representation capacity needed, but larger K may generalize poorly as well. A proper K should have enough representational capacity, but it should also generalize beyond the training data and even the training distribution.

4.4 Performance across Interaction Horizon

As the interaction horizon increases, the information the latent tokens should represent also increases. As such, a natural question arises: how does our method perform under different interaction horizons? Since **ALFWorld** has the longest interaction horizon, we mainly use **ALFWorld** for analysis in this section. To answer this question, we group completed trajectories in Table 1 by different interaction horizons and show the results in Figure 2b. We show the result for both $K = 32$ and ReAct in a side-by-side comparison. We want to understand whether the potential poor performance over long horizons is due to the model itself or due to our method.

As shown in Figure 2b, both methods' performance drops roughly linearly as the interaction horizon increases. However, in the long-horizon regime, our method does not worsen relative to ReAct. In the short-horizon regime, our method performs similarly to ReAct; in the long-horizon regime, it performs consistently better than ReAct. The performance drop in the long-horizon regime is due to the model or the agent's internal capabilities, not due to our method per se.

4.5 Ablation Study of Different Training Steps

Table 2: Ablation of different training stages. We compare behavior supervised learning (BSL), reinforcement learning (RL), and the full two-stage training approaches in **ALFWorld** when $K = 32$, aggregated between seen and unseen.

Training	ALFWorld
BSL only	55.6
RL only	51.4
BSL $\rightarrow$ RL	**63.8**

Finally, we conduct ablation studies on the contribution of each training stage. We compare the performance of the complete two-stage training approach versus behavior supervised learning only versus reinforcement learning only. Due to computational constraints, we report results only on **ALFWorld**.

Table 2 shows the results of different training stages. The BSL-only approach roughly matches ReAct. This shows that BSL successfully mimics ReAct's behavior. Interestingly, the reinforcement learning-only approach performs the worst. We conjecture that reinforcement learning provides little signal, especially in the early learning stage, for learning the context representation, and leads to training instability and suboptimal performance.

Overall, the two-stage training approach, which starts with behavior supervised learning followed by reinforcement learning, performs the best, and each training stage is essential.

5 Conclusion

In this study, we present $(\mathcal{LC})^2$: a framework that **l**earns **c**onstant-size **l**atent tokens for language model agents' **c**ontext. During each interaction step, a lightweight compressor maps the current turn's interaction and previous context into constant-size latent tokens. We freeze the language model and first train the compressor with behavior supervised learning, then fine-tune it with reinforcement learning using downstream task rewards. We evaluate our method on three representative benchmarks, and the results show that it significantly reduces token cost while increasing task performance. We further show that the number of latent tokens specifies a trade-off between capability and generalization, and that our method also works for long horizons. Further ablation studies also show that two training stages complement each other and neither is redundant.

AI use statement

In this work, we used generative AI tools to assist with writing proofs and implementing methods. We have not used generative AI tools to help develop theoretical models or conceptual frame-

works, formulate mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, support qualitative and thematic data analysis, or interpret results, and generate synthetic datasets, assist with translation, clean and reformat datasets, and provide critical ingredients for proving mathematical claims are not applicable to this work.

Additionally, we used generative AI tools to plot figures and survey the literature. We have reviewed all AI-assisted work. We checked that the LLM-generated code is correct. We checked that the LLM-generated writing is correct. We checked that the literature recommended by LLMs is not hallucinated. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

Reproducibility statement

Our code is available at `https://anonymous.4open.science/r/lc2-8F1E/`. We describe all details in Section 4.1 and Appendix B.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL `http://arxiv.org/abs/1607.06450`.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. xrag: Extreme context compression for retrieval-augmented generation with one token. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL `http://papers.nips.cc/paper_files/paper/2024/hash/c5cf13bfd3762821ef7607e63ee90075-Abstract-Conference.html`.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 3829–3846. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.232. URL `https://doi.org/10.18653/v1/2023.emnlp-main.232`.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL `https://doi.org/10.48550/arXiv.2501.12948`.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samual Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL `http://papers.nips.cc/paper_files/paper/2023/hash/5950bf290a1570ea401bf98882128160-Abstract-Datasets_and_Benchmarks.html`.

Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL `https://openreview.net/forum?id=uREj4ZuGJE`.

Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016. URL `http://arxiv.org/abs/1606.08415`.

Kelly Hong, Anton Troynikov, and Jeff Huber. Context rot: How increasing input tokens impacts llm performance. Technical report, Chroma, July 2025. URL `https://trychroma.com/research/context-rot`.

Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 32779–32798. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-LONG.1575. URL https://doi.org/10.18653/v1/2025.acl-long.1575.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A. Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. ACON: optimizing context compression for long-horizon LLM agents. *CoRR*, abs/2510.00615, 2025. doi: 10.48550/ARXIV.2510.00615. URL https://doi.org/10.48550/arXiv.2510.00615.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 5420–5438. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.276. URL https://doi.org/10.18653/v1/2025.emnlp-main.276.

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. *Trans. Assoc. Comput. Linguistics*, 12:157–173, 2024. doi: 10.1162/TACL_A_00638. URL https://doi.org/10.1162/tacl_a_00638.

Miao Lu, Weiwei Sun, Weihua Du, Zhan Ling, Xuesong Yao, Kang Liu, and Jiecao Chen. Scaling LLM multi-turn RL with end-to-end summarization-based context management. *CoRR*, abs/2510.06727, 2025. doi: 10.48550/ARXIV.2510.06727. URL https://doi.org/10.48550/arXiv.2510.06727.

Jesse Mu, Xiang Li, and Noah D. Goodman. Learning to compress prompts with gist tokens. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/3d77c6dcc7f143aa2154e7f4d5e22d68-Abstract-Conference.html.

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. *CoRR*, abs/2112.09332, 2021. URL https://arxiv.org/abs/2112.09332.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

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. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria,*

May 3-7, 2021. OpenReview.net, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Yu Su, Diyi Yang, Shunyu Yao, and Tao Yu. Language agents: Foundations, prospects, and risks. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pp. 17–24, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.emnlp-tutorials.3.

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. Scaling long-horizon LLM agent via context-folding. *CoRR*, abs/2510.11967, 2025. doi: 10.48550/ARXIV.2510.11967. URL https://doi.org/10.48550/arXiv.2510.11967.

Kimi Team. Kimi K3: open frontier intelligence. *CoRR*, abs/2607.24653, 2026. doi: 10.48550/ARXIV.2607.24653. URL https://doi.org/10.48550/arXiv.2607.24653.

Qwen Team. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Lei Wei, Xiao Peng, TT, Guannan Zhang, Chenhao Jiang, Hongyu Li, Lanbo Lin, Yuanwu Xu, Jiayao Liu, Kesu Wang, and Bin Wang. PACE: predictive adaptive context extraction for long-horizon LLM agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 27184–27199. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.1252. URL https://doi.org/10.18653/v1/2026.acl-long.1252.

Ronald J. Williams and Jing Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501, 1990. doi: 10.1162/neco.1990.2.4.490.

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, Zhongwang 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. *CoRR*, abs/2509.13313, 2025. doi: 10.48550/ARXIV.2509.13313. URL https://doi.org/10.48550/arXiv.2509.13313.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. 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, Brussels, Belgium, October 31 - November 4, 2018*, pp. 2369–2380. Association for Computational Linguistics, 2018. doi: 10.18653/V1/D18-1259. URL https://doi.org/10.18653/v1/d18-1259.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

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. *CoRR*, abs/2510.24699, 2025. doi: 10.48550/ARXIV.2510.24699. URL https://doi.org/10.48550/arXiv.2510.24699.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. Long context compression with activation beacon. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=1eQT9OzfNQ.

Yuwei Zhang, Chengyu Dong, Shuowei Jin, Changlong Yu, Hejie Cui, Hongye Jin, Xinyang Zhang, Hamed Bonab, Colin Lockard, Jianshu Chen, Zhenyu Shi, Jingbo Shang, Xian Li, and Bing Yin. Comem: Context management with A decoupled long-context model. *CoRR*, abs/2605.30842, 2026a. doi: 10.48550/ARXIV.2605.30842. URL https://doi.org/10.48550/arXiv.2605.30842.

Yuxiang Zhang, Jiangming Shu, Ye Ma, Xueyuan Lin, Shangxi Wu, and Jitao Sang. Memory as action: Autonomous context curation for long-horizon agentic tasks. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics, ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 19149–19164. Association for Computational Linguistics, 2026b. doi: 10.18653/V1/2026.FINDINGS-ACL.956. URL https://doi.org/10.18653/v1/2026.findings-acl.956.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 414–431. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.22. URL https://doi.org/10.18653/v1/2025.emnlp-main.22.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=oKn9c6ytLx.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. MEM1: learning to synergize memory and reasoning for efficient long-horizon agents. *CoRR*, abs/2506.15841, 2025. doi: 10.48550/ARXIV.2506.15841. URL https://doi.org/10.48550/arXiv.2506.15841.

A Algorithm Details

See Algorithm 1 for inference, Algorithm 2 for behavior supervised learning, and Algorithm 3 for GRPO reinforcement learning (Shao et al., 2024).

Algorithm 1 Inference Algorithm

Require: Compressor $\mathcal{C}_\phi$; $\mathcal{F}_\omega$ and $\mathcal{P}_\psi$; $\widehat{\mathbf{Z}}_{\text{inst}}$; frozen language model$\Pi_\theta$; environment ξ

Initialize recurrent state $\mathbf{Z}_1$
Receive initial observation o_1
for $t = 1, 2, \ldots, T$ **do**
    $\widetilde{\mathbf{Z}}_t \leftarrow \mathcal{P}_\psi(\mathbf{Z}_t)$
    $a_t \sim \Pi_\theta(\cdot \mid \widetilde{\mathbf{Z}}_t, o_t)$
    Execute a_t in ξ and receive o_{t+1}
    Construct interaction h_t
    if task terminates **then**
        break
    end if
    $\widehat{\mathbf{Z}}_t \leftarrow \mathcal{F}_\omega(\mathbf{Z}_t)$
    $\mathbf{Z}_{t+1} \leftarrow \mathcal{C}_\phi\left(\widehat{\mathbf{Z}}_t, h_t, \widehat{\mathbf{Z}}_{\text{inst}}\right)$
end for
return final agent output

Algorithm 2 Behavior Supervised Learning

Require: Successful trajectories $\mathcal{D}$; frozen language model Π_θ; pre-trained compressor $\mathcal{C}_\phi$

repeat
    Sample successful trajectory $\tau^* \sim \mathcal{D}$
    Initialize $\mathbf{Z}_1$
    $\mathcal{L}_{\text{BSL}} \leftarrow 0$
    for $t = 1, \ldots, T$ **do**
        $\widetilde{\mathbf{Z}}_t \leftarrow \mathcal{P}_\psi(\mathbf{Z}_t)$
        $\mathcal{L}_{\text{BSL}} \leftarrow \mathcal{L}_{\text{BSL}} - \log \Pi_\theta(a_t^* \mid \widetilde{\mathbf{Z}}_t, o_t)$
        Construct interaction h_t^*
        $\widehat{\mathbf{Z}}_t \leftarrow \mathcal{F}_\omega(\mathbf{Z}_t)$
        $\mathbf{Z}_{t+1} \leftarrow \mathcal{C}_\phi\left(\widehat{\mathbf{Z}}_t, h_t^*, \widehat{\mathbf{Z}}_{\text{inst}}\right)$
    end for
    $\Theta_{\text{BSL}} \leftarrow \Theta_{\text{BSL}} - \eta \nabla_{\Theta_{\text{BSL}}} \mathcal{L}_{\text{BSL}}$
until convergence
return Θ_{BSL}

Algorithm 3 GRPO Training

Require: Frozen language model Π_θ; BSL-initialized parameters Θ^*_{BSL}; group size G; clipping coefficient ε; TBPTT horizon M_{TBPTT}

repeat
  Sample task instance x
  $\Theta^{\mathrm{old}}_{\mathrm{GRPO}} \leftarrow \Theta_{\mathrm{GRPO}}$
  for $i = 1, \ldots, G$ **do**
    Initialize $\mathbf{Z}^{\mathrm{old}}_{i,1}$
    for $t = 1, \ldots, T_i$ **do**
      $\widetilde{\mathbf{Z}}^{\mathrm{old}}_{i,t} \leftarrow \mathcal{P}_{\psi^{\mathrm{old}}}(\mathbf{Z}^{\mathrm{old}}_{i,t})$
      Sample $a_{i,t} \sim \pi_\theta(\cdot \mid \widetilde{\mathbf{Z}}^{\mathrm{old}}_{i,t}, o_{i,t})$
      Execute $a_{i,t}$ and construct $h_{i,t}$
      if task terminates **then**
        break
      end if
      $\widehat{\mathbf{Z}}^{\mathrm{old}}_{i,t} \leftarrow \mathcal{F}_{\omega^{\mathrm{old}}}(\mathbf{Z}^{\mathrm{old}}_{i,t})$
      $\mathbf{Z}^{\mathrm{old}}_{i,t+1} \leftarrow \mathcal{C}_{\phi^{\mathrm{old}}}\left(\widehat{\mathbf{Z}}^{\mathrm{old}}_{i,t}, h_{i,t}, \widehat{\mathbf{Z}}^{\mathrm{old}}_{\mathrm{inst}}\right)$
    end for
    Obtain trajectory reward R_i
  end for
  $\mu_R \leftarrow \frac{1}{G}\sum_{i=1}^{G} R_i$
  $\sigma_R \leftarrow \sqrt{\frac{1}{G}\sum_{i=1}^{G}(R_i - \mu_R)^2}$
  for $i = 1, \ldots, G$ **do**
    $\hat{A}_i \leftarrow (R_i - \mu_R)/(\sigma_R + \delta)$
  end for
  Recompute $\mathbf{Z}^{\Theta_{\mathrm{GRPO}}}_{i,t}$ under current Θ_{GRPO} using TBPTT horizon M_{TBPTT}
  for all generated tokens $y_{i,t,l}$ **do**
    $p_{\mathrm{new}} \leftarrow \Pi_\theta(y_{i,t,l} \mid \mathcal{P}_\psi(\mathbf{Z}^{\Theta_{\mathrm{GRPO}}}_{i,t}), o_{i,t}, y_{i,t,<l})$
    $p_{\mathrm{old}} \leftarrow \Pi_\theta(y_{i,t,l} \mid \mathcal{P}_{\psi^{\mathrm{old}}}(\mathbf{Z}^{\mathrm{old}}_{i,t}), o_{i,t}, y_{i,t,<l})$
    $r_{i,t,l} \leftarrow p_{\mathrm{new}}/p_{\mathrm{old}}$
    $\mathcal{J}_{i,t,l} \leftarrow \min\left(r_{i,t,l}\widehat{A}_i,\ \mathrm{clip}(r_{i,t,l}, 1-\varepsilon, 1+\varepsilon)\widehat{A}_i\right)$
  end for
  $\mathcal{J}_{\mathrm{GRPO}} = \underset{i,t,l}{\mathrm{mean}}\left[\mathcal{J}_{i,t,l}\right]$
  $\Theta_{\mathrm{GRPO}} \leftarrow \Theta_{\mathrm{GRPO}} + \eta\nabla_{\Theta_{\mathrm{JRPO}}}\mathcal{J}_{\mathrm{GRPO}}$
until convergence
return Θ_{GRPO}

B ADDITIONAL DETAILS

B.1 EXAMPLE PROMPTS

Below is the prompt template for each benchmark.

B.1.1 ALFWORLD

ALFWorld Prompt Template

Task
{task_instruction}
Observation
{current_observation}
Complete the task by interacting with the environment. At each step, reason about what you should do next and then issue one valid action.
Use the following format:
```
Thought: <reasoning>
Action: <environment action>
```

B.1.2 WEBSHOP

WebShop Prompt Template

Instruction
{shopping_instruction}
Observation
{current_webpage}
Find and purchase the product that best satisfies the user's request. At each step, reason about the current webpage and select one valid action.
Available actions include:
```
search[<query>]
click[<clickable item>]
```
Use the following format:
```
Thought: <reasoning>
Action: <valid action>
```

B.1.3 HOTPOTQA

HotpotQA Prompt Template

Question
{question}
Answer the question by gathering the necessary evidence. The question may require combining information from multiple sources.
At each step, either search for additional information or provide the final answer.
Use the following format:
```
Thought: <reasoning>
Action: Search[<query>]
```
or, when sufficient evidence has been collected,
```
Thought: <reasoning>
Action: Finish[<answer>]
```

B.2 MODELS

For the frozen language model, we use QWEN2.5-7B-INSTRUCT. For the pre-trained compressor, we use QWEN2.5-0.5B.

Table 3: Training hyperparameters used in the main experiments.

Hyperparameter	Value
Optimizer	AdamW
Learning rate (BSL)	5e-5
Learning rate (RL)	1e-6
Weight decay	0.01
Training epochs	1
LR scheduler	Cosine
TBPTT length	8
GRPO group size	8
KL coefficient	0.04

B.3 Evaluation Details

We run three independent trials and report the mean. For **HotpotQA**, we randomly sample 1500 instances instead of the full benchmark. We use greedy decoding for all evaluations.

B.4 Implementation and Training Details

We run all the experiments on Google Colab using a single NVIDIA A100 80GB GPU. We implement the method using PyTorch and TRL. Table 3 lists our training hyperparameters.

B.5 Baselines

For the sliding window baseline, we maintain the most recent 4 steps of information. For the summarization, we use the following prompt:

Summarization Prompt

Instruction: Summarize the interaction history below for the agent.
Preserve:

1. the task goal and constraints,
2. important observations and discovered information,
3. actions already taken and their outcomes,
4. current progress and state,
5. unresolved subgoals or remaining steps,
6. failures or actions that should not be repeated.

Remove redundant details and verbose reasoning. Do not add information that is not present in the history. Keep the summary concise but include enough information for the agent to continue the task without access to the original history.

Interaction History:
{`history`}
Summary: