ZipAct: Zipping Interaction History into a Compact State for Efficient LLM Agents

Zhiming Pan
Peking University
thomas_p@stu.pku.edu.cn

Xiao Luo
University of Wisconsin–Madison
xiao.luo@wisc.edu

Abstract

Most existing LLM agents, such as ReAct (Yao et al., 2023), rely on the entire interaction history to make decisions. However, this paradigm suffers from the "Context Snowball" effect: as the task progresses, the history grows unboundedly, which not only consumes excessive tokens but also dilutes the agent's attention. To address this issue, we propose **ZipAct**, a lightweight framework that "zips" the lengthy history into a compact state. Instead of feeding the full history to the model, ZipAct maintains a structured state table, comprising the agent's goal, world status and constraints, and updates it dynamically at each step. This design shifts agent reasoning from a history-dependent to a state-dependent paradigm, significantly reducing computational cost from quadratic ($O(T^2)$) to linear ($O(T)$). Extensive experiments across ALFWorld, SciWorld, and WebShop demonstrate that ZipAct drastically reduces token usage while preserving or improving success rates compared to strong baselines.[1]

1 Introduction

Large Language Model (LLM) agents have demonstrated remarkable potential in solving complex, long-horizon tasks (Forootani, 2025), ranging from embodied activities (Shridhar et al., 2021) to scientific reasoning (Wang et al., 2022) and web navigation (Yao et al., 2022). However, these tasks require extensive multi-turn interactions, leading to significant context growth. Consequently, a fundamental challenge arises: how to enable agents to handle unboundedly growing interaction histories within finite context windows without compromising efficiency or accuracy.

Prevalent frameworks like ReAct (Yao et al., 2023) use the entire interaction history as context, concatenating all past observations and actions as input at each step. Even recent advancements like ReflAct (Kim et al., 2025) and StateAct (Rozanov and Rei, 2024), which introduce self-reflection or state descriptions, typically append these features to the ever-growing history. While effective for short trajectories, these "append-only" approaches suffer from a critical phenomenon we term *"Context Snowball"*, creating a dual bottleneck (see Figure 4). First, the cumulative token consumption increases quadratically with the number of turns T, making long-horizon tasks prohibitively expensive. Second, the accumulation of raw history introduces substantial noise; irrelevant details from early steps often obscure critical information, inducing hallucination or losing track of the immediate sub-goal (Liu et al., 2024). Although attempts like token masking (Lindenbauer et al., 2025) or compression via distillation (Kang et al., 2025a) have been proposed, they either risk losing critical information or require expensive additional training. This necessi-

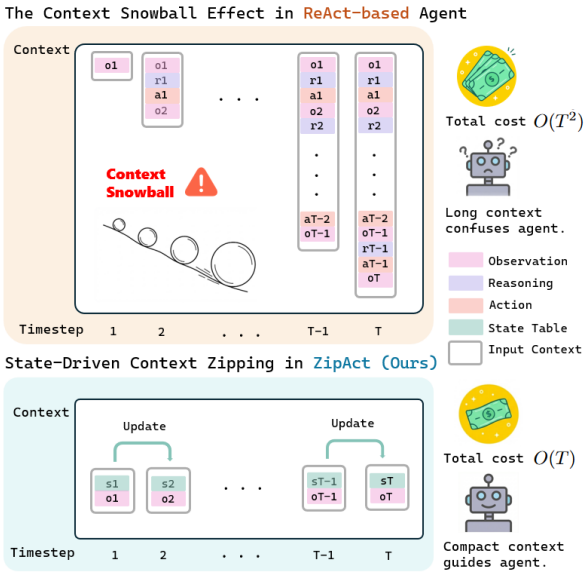

Figure 1: **ReAct vs. ZipAct.** ReAct accumulates raw history, leading to a "Context Snowball" ($O(T^2)$). ZipAct zips history into a compact state ($O(T)$), effectively eliminating redundancy and distraction.

[1]Implementation code is available at: https://github.com/your_username/ZipAct

tates a fundamental shift from passively recording history to actively managing state.

To bridge this gap, we propose **ZipAct**, a lightweight and training-free framework that shifts agent reasoning from a history-dependent to a state-dependent paradigm. ZipAct maintains a compact structured state S_t that "zips" the verbose history into three components: the *Goal State* ($\mathcal{G}$) for tracking hierarchical task progress, the *World State* ($\mathcal{W}$) for abstracting key environmental observations, and the *Constraint State* ($\mathcal{C}$) for managing action validity and feedback. Structurally, we decouple the system into a memory-less *Actor*, which decides actions based solely on the current state S_t, and a *State Updater*, which synthesizes the semantic transition from t to $t+1$. This architecture ensures that the input context length remains bounded and decoupled from the interaction history, effectively reducing the cumulative inference cost from quadratic ($O(T^2)$) to linear ($O(T)$).

We evaluate ZipAct using both open-source (Qwen-2.5-7B/32B-Instruct (Yang et al., 2024a)) and proprietary (GPT-4o-mini/4o (OpenAI, 2024)) LLMs. We conduct extensive experiments across three benchmarks: ALFWorld (Shridhar et al., 2021), SciWorld (Wang et al., 2022), and WebShop (Yao et al., 2022). Empirical results demonstrate that ZipAct effectively eliminates the "Context Snowball" effect. Specifically, ZipAct reduces token consumption by **XX%–XX%** across datasets while preserving or even improving the success rate compared to full-history baselines. With linear complexity ($O(T)$), ZipAct provides a robust and scalable alternative to accumulating raw history.

The contribution of our work can be summarized as follows: (1) We propose ZipAct, a training-free framework that resolves the "Context Snowball" bottleneck by transitioning agents to a state-dependent paradigm. (2) We design a unified Goal–World-Constraint schema that decouples inference cost from interaction history, achieving linear complexity ($O(T)$). (3) Extensive experiments demonstrate that ZipAct significantly reduces token consumption while maintaining robust success rates across diverse benchmarks.

2 Related Work

2.1 State-based Agent Reasoning

Drawing inspiration from Chain-of-Thought reasoning (Wei et al., 2022), ReAct (Yao et al., 2023) introduces a framework that interleaves reasoning traces with actions. Subsequent enhancements seek to improve this backbone through various mechanisms: Reflexion (Shinn et al., 2023) incorporates iterative self-reflection on failures; HiPlan (Li et al., 2025), HyperTree (Gui et al., 2025), and HIMA (Ahn et al., 2025) leverage hierarchical planning and milestone-level guidance; and RAFA (Liu et al., 2023) evaluates possible future trajectories. Specifically, ReflAct (Kim et al., 2025) and StateAct (Rozanov and Rei, 2024) focus on explicit state tracking to ensure goal alignment and world-grounding. However, these approaches append state or reflection summaries to an ever-growing trajectory. This reliance on cumulative history leads to the "Context Snowball" effect. In contrast, ZipAct shifts the paradigm to a state-dependent approach, demonstrating that a compact, structured state is sufficient for robust long-horizon reasoning.

2.2 Context Management for LLM Agents

Various strategies have been explored to manage the computational overhead of long-context agents. Heuristic mechanisms, such as token masking (Research, 2025a), trajectory pruning (Research, 2025b), and observation truncation in SWE-agent (Yang et al., 2024b), aim to reduce costs but often risk losing critical information. More sophisticated solutions include context distillation like ACON (Kang et al., 2025b), memory-augmented systems such as MemGPT (Packer et al., 2023), LightMem (Fang et al., 2025), and GAM (Yan et al., 2025), or test-time optimization via plan caching (Zhang et al., 2025) and action compression (Anonymous, 2026). While these methods are often ad-hoc or require expensive model-specific training (white-box dependence), ZipAct provides a semantic, training-free state update mechanism. It ensures a constant $O(1)$ per-step context size and linear $O(T)$ cumulative complexity, serving as a plug-and-play efficiency booster for LLM agents.

3 Preliminaries

We formulate the agent-environment interaction as a Partially Observable Markov Decision Process (POMDP), defined as $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{R}, \mathcal{G} \rangle$, where $\mathcal{S}$ denotes the latent state space (e.g., the full status of a household), which is not fully visible to the agent. $\mathcal{A}$ is the discrete action space (e.g., `open drawer 1`). $\mathcal{O}$ is the observation space, where $o_t \in \mathcal{O}$ provides a partial view of the state s_t at step t. $\mathcal{P}$ represents the transition dynamics, and

$\mathcal{R}$ is the reward function. $\mathcal{G}$ is the natural language goal (e.g., "put a clean lettuce in dining table"). The agent's objective is to execute a sequence of actions that successfully completes the task $\mathcal{G}$ within a limited number of steps.

The ReAct Paradigm. The dominant framework for LLM agents, exemplified by ReAct (Yao et al., 2023) and its numerous variants (e.g., ReflAct (Kim et al., 2025)), relies on *interleaved reasoning and acting*. To facilitate planning, these methods introduce a *Thought Space* $\mathcal{T}$, where a thought $\tau_t \in \mathcal{T}$ is a natural language reasoning trace generated before the action. At each time step t, the agent's decision is conditioned on the entire history H_t, which accumulates all past observations, thoughts, and actions:

$$H_t = [\mathcal{G}, o_0, \tau_0, a_0, o_1, \tau_1, a_1, \ldots, o_t]. \quad (1)$$

The policy π operates in a cycle: first generating a thought $\tau_t \sim \pi(\cdot|H_t)$, appending it to the context, and then predicting an action $a_t \sim \pi(\cdot|H_t, \tau_t)$.

While this paradigm effectively grounds the model's reasoning, Eq. 1 reveals a critical structural flaw: the input context H_t grows linearly with t. As we analyze in the next section, this *context snowballing* introduces quadratic computational complexity and accumulates distractive noise, fundamentally limiting the efficiency and robustness of long-horizon agents.

4 The Context Snowball Challenge

In this section, we analyze the limitations of history-dependent architectures (e.g., ReAct (Yao et al., 2023)) from two perspectives: computational efficiency and cognitive distraction. We term the phenomenon where the input context grows uncontrollably as *Context Snowballing*.

4.1 Quadratic Complexity

The standard practice of concatenating all past observations, thoughts, and actions imposes prohibitive costs in long-horizon tasks. Let L_{step} denote the average number of tokens generated in a single interaction cycle. For a task with horizon T, the length of the input context at step t is $|C_t| \approx t \cdot L_{\text{step}}$. Consequently, the cumulative token consumption $\mathcal{C}_{total}$ scales quadratically with the task horizon:

$$\mathcal{C}_{total} \approx \sum_{t=1}^{T} (t \cdot L_{\text{step}}) = O(T^2) \quad (2)$$

Figure 2: **The Cost of Snowballing.** Cumulative token usage comparison between ReAct (Orange) and ZipAct (Blue) on a standard ALFWorld task. ReAct exhibits quadratic $O(T^2)$ growth, while ZipAct maintains linear $O(T)$ efficiency.

Figure 3: **Cognitive Noise.** A motivating example showing how critical information (e.g., "Key is on the table") becomes buried in the middle of a 40-step history, causing the agent to hallucinate that it hasn't found the key.

This quadratic growth is unsustainable. As illustrated in Figure 2 (orange line), the token usage of a ReAct agent explodes as the task progresses. For a 50-step task in ALFWorld, ReAct processes over [X]k tokens in total, whereas a linear method would only require [Y]k. This results in high latency and deployment costs.

4.2 Cognitive Load: Information Dilution

Beyond computational costs, the "snowballing" history imposes a severe cognitive burden. In realistic environments, agents frequently execute long trajectories involving exploration and trial-and-error. However, ReAct-based frameworks *indiscriminately retain* every interaction detail, regardless of relevance. Consequently, critical signals required for correct reasoning (e.g., the original goal and current inventory) become increasingly obscured by an accumulation of redundant information. This aligns with the "Lost-in-the-Middle" phenomenon (Liu et al., 2024), where LLMs struggle to retrieve key details from extended contexts. We argue that this unstructured accumulation dilutes the agent's focus. As illustrated in Figure 3, as history grows, the model's attention to the initial goal diminishes, often precipitating "hallucination loops" where the agent repeats invalid actions.

5 Proposed Method: ZipAct

To mitigate the *context snowballing* effect inherent in history-dependent agents (e.g., ReAct), we introduce **ZipAct**. The core insight is to shift the agent's reasoning from a *history-dependent* paradigm to a *state-dependent* paradigm.

Formally, rather than conditioning the policy on the ever-growing interaction trajectory H_t, ZipAct maintains a compact, structured state S_t that explicitly tracks the task progress and environment status. The decision process is decoupled into two phases:

state updating and action generation.

$$S_t = \mathcal{U}(S_{t-1}, a_{t-1}, o_t) \quad (3)$$

$$\tau_t, a_t \sim \pi(\cdot|S_t, o_t) \quad (4)$$

where $\mathcal{U}$ is the State Updater and π is the Actor policy. Since the dimension of S_t is designed to be fixed (or bounded) regardless of step t, the input length for the Actor remains constant ($O(1)$ context). Consequently, this architectural shift reduces the cumulative computational complexity from quadratic $O(T^2)$ to linear $O(T)$, making long-horizon reasoning both efficient and robust.

5.1 Overview

Following the principle of modular system design, ZipAct decouples information synthesis from action execution. This architecture comprises two collaborative units. The *Actor (π)* operates as a memory-less decision engine. Unlike standard agents that re-read the full history log, the Actor predicts the next action a_t conditioned *solely* on the current synthesized state S_t and the immediate observation o_t. The *State Updater ($\mathcal{U}$)*, acting as a semantic compressor, is responsible for the transition dynamics. It digests the raw feedback from the environment and "zips" the interaction history into a refreshed state S_t, explicitly filtering out noise while preserving actionable context.

5.2 Structured State Representation

A key challenge in discarding history is preserving the critical context required for valid reasoning. To address this, we define the state as a structured tuple $S_t = \langle \mathcal{G}_t, \mathcal{W}_t, \mathcal{C}_t \rangle$, where each component targets a specific dimension of the POMDP.

Goal State ($\mathcal{G}_t$). This component maintains the hierarchical consistency of the mission. It is composed of a static global instruction g_{global}, a dynamic sub-goal queue Q_{plan}, and the current objective g_{curr}. By explicitly isolating g_{curr}, ZipAct ensures the Actor remains grounded in the immediate step, while Q_{plan} prevents the agent from losing track of the long-term objective.

World State ($\mathcal{W}_t$). Serving as a denoised environment snapshot, $\mathcal{W}_t$ abstracts raw textual observations into task-relevant semantic variables. It tracks the agent's logical location l_t, inventory status I_t, and a selective entity map O_{map} (e.g., {"Fridge": "Open"}). Unlike raw history which accumulates irrelevant descriptions, $\mathcal{W}_t$ only retains state changes that are causally relevant to future actions.

Algorithm 1 ZipAct Inference Loop

Require: Instruction $\mathcal{I}$, Env $\mathcal{E}$
Init: $S_0 \leftarrow \langle \mathcal{I}, \emptyset, \emptyset \rangle$; $o_0 \leftarrow \mathcal{E}$.reset()
for $t = 1$ **to** T_{max} **do**
  $\tau_t, a_t \leftarrow \pi(S_{t-1}, o_{t-1})$ *// Act on $O(1)$ state*
  $o_t, \_, done \leftarrow \mathcal{E}$.step($a_t$)
  if $done$ **then**
    return success
  end if
  $\Delta_g, \Delta_w, \Delta_c \leftarrow \mathcal{U}(S_{t-1}, a_t, o_t)$
  $S_t \leftarrow$ Update($S_{t-1}, \langle \Delta_g, \Delta_w, \Delta_c \rangle$)
end for
return failure

Constraint State ($\mathcal{C}_t$). To mitigate the risk of cyclic failures common in memory-less systems, $\mathcal{C}_t$ serves as an explicit anti-loop mechanism. It records discovered negative constraints (e.g., "Drawer 1 is locked") and tracks visited states. This effectively prunes the invalid search space without requiring the model to re-analyze past failure logs.

5.3 State Update Mechanism

Maintaining the state S_t requires a mechanism to synthesize new information while adhering to the bounded context size. We instantiate the Updater $\mathcal{U}$ using an agent functioning as a *Semantic Compiler*. At each time step t, the Updater performs a structured transformation defined as $S_t \leftarrow \mathcal{U}(S_{t-1}, a_{t-1}, o_t)$.

To ensure robustness, the update process follows a strict schema covering three logical flows. First, *Goal Progression Analysis* evaluates whether the observation o_t entails the completion of g_{curr}. If the entailment holds (e.g., observing "heat" after "turning on microwave"), g_{curr} is marked complete and removed from Q_{plan}; otherwise, the goal persists. Simultaneously, *World Patching* extracts semantic changes—such as item acquisition or location shifts—to overwrite the corresponding slots in $\mathcal{W}_{t-1}$, discarding non-essential descriptive noise. Finally, *Failure Reflection* detects execution errors. If o_t indicates a failure, the specific cause is parsed into a persistent rule in $\mathcal{C}_t$.

Crucially, this mechanism ensures that the size of S_t is determined by the problem complexity (e.g., number of active objects) rather than the trajectory length T. Unlike ReAct, where the context scales as $|H_t| \propto T$, ZipAct maintains $|S_t| \approx O(1)$, thereby guaranteeing linear cumulative inference cost $O(T)$ for long-horizon tasks.

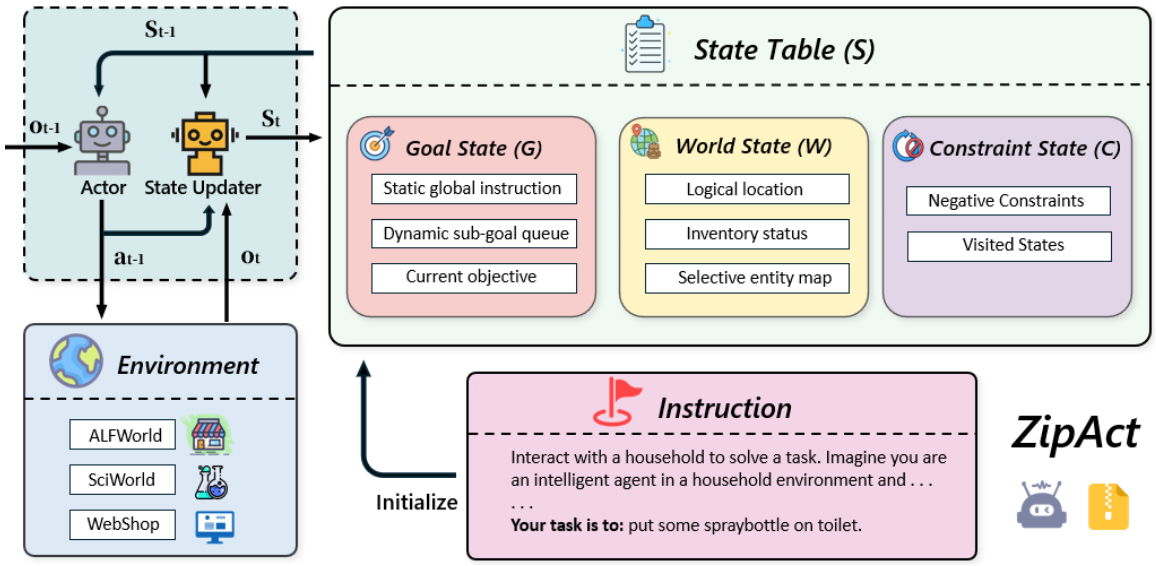

Figure 4: The overall framework of the proposed ZipAct. The State Updater synthesizes the interaction history into a compact State Table comprising Goal (G), World (W), and Constraint (C) states. The Actor generates actions based on this structured state, effectively decoupling inference cost from trajectory length.

6 Experiments

6.1 Experimental Settings

Benchmarks. We benchmark ZipAct on three challenging text-based environments: ALFWorld (Shridhar et al., 2021), SciWorld (Wang et al., 2022), and WebShop (Yao et al., 2022). ALFWorld evaluates embodied agents on household tasks requiring multi-step planning, SciWorld assesses scientific reasoning in complex interactive laboratory scenarios, and WebShop tests agents on web navigation and information seeking in a simulated e-commerce platform. For evaluation, ALFWorld uses binary task success, while SciWorld and WebShop provide granular reward signals, allowing evaluation based on both success rate and average score (0-100).Additional benchmark details are provided in Appendix C.

Agent Models. We use GPT-4o and GPT-4o-mini (OpenAI, 2024), and Qwen-2.5-7B/32B-Instruct (Yang et al., 2024a) as underlying models. GPT-4o variants serve as proprietary models, while Qwen-2.5 variants represent open-source counterparts, each with large and small sizes.

6.2 Comparison with Prior Methods

We compare ZipAct with four baselines. Implementation details are in Appendix C.

(1) ReAct (Yao et al., 2023): The standard approach conditioning decisions on the entire raw interaction history at each step.

(2) ReAct + Observation Masking (Lindenbauer et al., 2025): A strong heuristic baseline that retains the full action history but masks older observations to reduce token usage.

(3) ReAct + Summarization: (Park et al., 2023; Zhong et al., 2023): Represents compression methods where the agent summarizes interaction history into natural language to replace raw context.

(4) Reflexion (Shinn et al., 2023): The agent generates and appends verbal self-reflections upon failure, representing methods that augment context to improve reasoning.

Results.

6.3 Further Analysis.

Ablation Study. To emphasize the effectiveness of our structured state design, we compare three variants of ZipAct: (1) V1 (Goal Only), which removes both World State and Constraint State. (2) V2 (w/o Constraint), which removes the Constraint State but retains goal and world tracking. (3) ZipAct (Full Model), which utilizes the unified $\langle \mathcal{G}, \mathcal{W}, \mathcal{C} \rangle$ tuple. We show the compared performance in Table **??**. From the results, we have the following observations. Firstly, by comparing V1 and V2, we observe that the lack of world grounding ($\mathcal{W}$) leads to a significant drop in success rate, indicating that the agent requires an explicit memory of environmental changes to avoid hallucination. Secondly, ZipAct outperforms V2, particularly in long-horizon tasks. This validates the importance of the Constraint State ($\mathcal{C}$) as an anti-loop mechanism, preventing the agent from repeating invalid actions. Overall, the complete

Table 1: Performance comparison of ZipAct with ReAct, Mask Obs., Summary, and Reflexion across ALFWorld, ScienceWorld, and WebShop. SR denotes success rate. Token Reduction indicates the average percentage of tokens saved compared to the ReAct baseline.

Model	Method	Success Rate (SR) %			Average SR	Token Reduction
		ALFWorld	Science	WebShop		
GPT-4o-mini	ReAct	53.0	14.2	48.5	38.6	-
	Mask Obs.	*To Run*	*To Run*	*To Run*	-	-15.2%
	Summary	*To Run*	*To Run*	*To Run*	-	-10.5%
	Reflexion	66.4	18.5	55.2	46.7	+120%
	ZipAct	**Your Data**	**Your Data**	**Your Data**	**-**	**-68.4%**
GPT-4o	ReAct	85.1	30.5	62.4	59.3	-
	Mask Obs.	*To Run*	*To Run*	*To Run*	-	-15.2%
	Summary	*To Run*	*To Run*	*To Run*	-	-10.5%
	Reflexion	93.3	42.0	70.1	68.5	+120%
	ZipAct	**Your Data**	**Your Data**	**Your Data**	**-**	**-70.9%**
Qwen-2.5-7B	ReAct	61.5	8.5	42.1	37.4	-
	Mask Obs.	*To Run*	*To Run*	*To Run*	-	-15.2%
	Summary	*To Run*	*To Run*	*To Run*	-	-10.5%
	Reflexion	68.2	10.2	46.0	41.5	+120%
	ZipAct	**Your Data**	**Your Data**	**Your Data**	**-**	**-65.2%**
Qwen-2.5-32B	ReAct	81.7	26.4	58.9	55.7	-
	Mask Obs.	*To Run*	*To Run*	*To Run*	-	-15.2%
	Summary	*To Run*	*To Run*	*To Run*	-	-10.5%
	Reflexion	89.5	35.1	64.2	62.9	+120%
	ZipAct	**Your Data**	**Your Data**	**Your Data**	**-**	**-63.8%**

G-W-C schema provides the most robust guidance with minimal token overhead.

6.4 Case Study

7 Conclusion

We proposed ZipAct, a state-dependent framework that effectively addresses the "Context Snowball" bottleneck in LLM agents. While ReAct relies on accumulating raw history, it suffers from quadratic computational costs ($O(T^2)$) and attention dilution. In contrast, ZipAct zips verbose history into a compact, structured state representation, achieving linear complexity ($O(T)$) and focused reasoning. Our experiments demonstrate that ZipAct drastically reduces token usage while preserving high success rates across diverse benchmarks. We hope this work inspires further research into efficient context management for complex long-horizon tasks.

Limitations

Despite the efficacy of our proposed ZipAct framework in achieving linear complexity and state-dependent reasoning, there are limitations inherent to our design that warrant further discussion.

First, the "zipping" process creates an information bottleneck. We assume the State Updater can perfectly distinguish signal from noise. However, in long-horizon tasks, details discarded early on might become important later. Unlike full-history models, ZipAct cannot easily backtrack. Future work could explore how to selectively recall raw history when the current state is inadequate.

Second, our structured state schema relies on heuristic design. The current $\langle \mathcal{G}, \mathcal{W}, \mathcal{C} \rangle$ tuple is tailored for embodied or web agents. This structure may lack the flexibility to capture nuances in open-ended domains like creative writing. How to enable agents to automatically discover optimal state representations without human-defined schemas remains an intriguing direction for future research.

Acknowledgments

References

Daechul Ahn, San Kim, and Jonghyun Choi. 2025. Society of mind meets real-time strategy: A hierarchical multi-agent framework for strategic reasoning. *arXiv preprint arXiv:2508.06042*.

Anonymous. 2026. Os-catalyst: Advancing computer-

using agents efficiency through adaptive action compression. *ICLR Submission*.

Jizhan Fang, Xinle Deng, Haoming Xu, Ningyu Zhang, and 1 others. 2025. Lightmem: Lightweight and efficient memory-augmented generation. *arXiv preprint arXiv:2510.18866*.

Ali Forootani. 2025. A survey on mathematical reasoning and optimization with large language models. *arXiv preprint arXiv:2503.17726*.

Runquan Gui, Zhihai Wang, Jie Wang, Chi Ma, Huiling Zhen, Mingxuan Yuan, Jianye Hao, Defu Lian, Enhong Chen, and Feng Wu. 2025. Hypertree planning: Enhancing llm reasoning via hierarchical thinking. *arXiv preprint arXiv:2505.02322*.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. 2025a. Acon: Optimizing context compression for long-horizon llm agents. *arXiv preprint arXiv:2510.00615*.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. 2025b. Acon: Optimizing context compression for long-horizon llm agents. *arXiv preprint arXiv:2510.00615*.

Jeonghye Kim, Sojeong Rhee, Minbeom Kim, Dohyung Kim, Sangmook Lee, Youngchul Sung, and Kyomin Jung. 2025. Reflact: World-grounded decision making in llm agents via goal-state reflection. *arXiv preprint arXiv:2505.15182*.

Ziyue Li, Yuan Chang, Gaihong Yu, and Xiaoqiu Le. 2025. Hiplan: Hierarchical planning for llm-based agents with adaptive global-local guidance. *arXiv preprint arXiv:2508.19076*.

Tobias Lindenbauer, Igor Slinko, Ludwig Felder, Egor Bogomolov, and Yaroslav Zharov. 2025. The complexity trap: Simple observation masking is as efficient as llm summarization for agent context management. *arXiv preprint arXiv:2508.21433*.

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

Zeyu Liu and 1 others. 2023. Rafa: Reasoning and acting with future awareness for llm agents. *arXiv preprint arXiv:2305.14325*.

OpenAI. 2024. Gpt-4o system card. OpenAI Model Release.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, and 1 others. 2023. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pages 1–22.

JetBrains Research. 2025a. Improving the efficiency of llm agent systems through trajectory reduction. *NeurIPS Workshop on Deep Learning for Code in the Agentic Era*.

JetBrains Research. 2025b. Improving the efficiency of llm agent systems through trajectory reduction. *arXiv preprint arXiv:2508.21433*.

Nikolai Rozanov and Marek Rei. 2024. Stateact: State tracking and reasoning for acting and planning with large language models. *arXiv e-prints*, pages arXiv–2410.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. {ALFW}orld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11279–11298.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

B.Y. Yan, Chaofan Li, Hongjin Qian, and Zheng Liu. 2025. General agentic memory via deep research. *arXiv preprint arXiv:2510.27418*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2024a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilic Kilinc, Shunyu Lawrence, Karthik Narasimhan, and Ofir Press. 2024b. Swe-agent: Agent-computer interfaces enable multi-turn browsing and code editing. *arXiv preprint arXiv:2405.15793*.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Qizheng Zhang, Michael Wornow, and Kunle Olukotun. 2025. Agentic plan caching: Test-time memory for fast and cost-efficient llm agents. *NeurIPS*.

Wanjun Zhong, Lianghong Liang, Iz Zedan, Alessandra Wang, and 1 others. 2023. Memorybank: Enhancing large language models with long-term memory. *arXiv preprint arXiv:2305.10250*.

A Example Appendix

This is an appendix.