DeepAgent: A General Reasoning Agent with Scalable Toolsets

Xiaoxi Li[1,2*], Wenxiang Jiao[2], Jiarui Jin[2], Guanting Dong[1], Jiajie Jin[1], Yinuo Wang[2], Hao Wang[2], Yutao Zhu[1], Ji-Rong Wen[1], Yuan Lu[2], Zhicheng Dou[1]

[1]Renmin University of China [2]Xiaohongshu Inc.

{xiaoxi_li, dou}@ruc.edu.cn, luyuan3@xiaohongshu.com

Abstract

Large reasoning models have demonstrated strong problem-solving abilities, yet real-world tasks often require external tools and long-horizon interactions. Existing agent frameworks typically follow predefined workflows, which limit autonomous and global task completion. In this paper, we introduce **DeepAgent**, an end-to-end deep reasoning agent that performs autonomous thinking, tool discovery, and action execution within a single, coherent reasoning process. To address the challenges of long-horizon interactions, particularly the context length explosion from multiple tool calls and the accumulation of interaction history, we introduce an autonomous memory folding mechanism that compresses past interactions into structured episodic, working, and tool memories, reducing error accumulation while preserving critical information. To teach general-purpose tool use efficiently and stably, we develop an end-to-end reinforcement learning strategy, namely ToolPO, that leverages LLM-simulated APIs and applies tool-call advantage attribution to assign fine-grained credit to the tool invocation tokens. Extensive experiments on eight benchmarks, including general tool-use tasks (ToolBench, API-Bank, TMDB, Spotify, ToolHop) and downstream applications (ALFWorld, WebShop, GAIA, HLE), demonstrate that DeepAgent consistently outperforms baselines across both labeled-tool and open-set tool retrieval scenarios. This work takes a step toward more general and capable agents for real-world applications. The code and demo are available at **https://github.com/RUC-NLPIR/DeepAgent**.

Keywords

Large Reasoning Models, Autonomous Agents, Tool Retrieval, Memory Mechanism, Reinforcement Learning

1 Introduction

The rapid advancement of large language models (LLMs) has inspired the development of LLM-powered agents, which have found broad applications in scenarios such as web information seeking, software engineering, and personal assistance [19, 39, 53]. Existing agent frameworks predominantly rely on predefined workflows, exemplified by methods like ReAct [67] and Plan-and-Solve [54], which employ structured planning processes and iterative "Reason-Act-Observe" cycles as illustrated in Figure 2(a). Although effective in simpler tasks, these approaches suffer from several critical limitations: (1) lack of autonomy in execution steps and overall procedure; (2) inability to dynamically discover tools during task execution; (3) deficiency in fully autonomous management of interactive memory; and (4) insufficient depth and coherence in reasoning about the entire task. These fundamental shortcomings severely constrain the agents' ability to tackle real-world problems, particularly for complex tasks that demand general tool-use and long-horizon interaction with the environment.

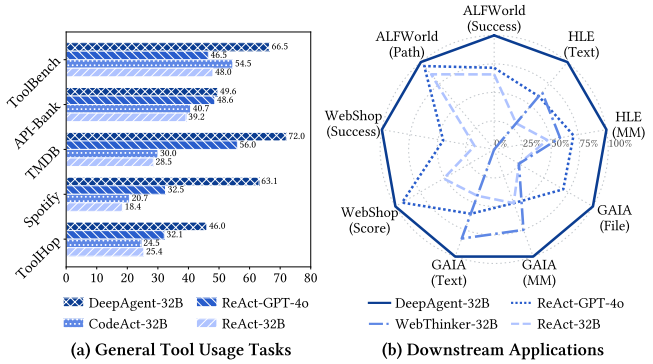


Figure 1: Overall performance on (a) general tool usage tasks and (b) downstream applications (best score as 100%).

Recently, the advent of large reasoning models (LRMs) has demonstrated the capability to solve complex problems in domains like mathematics, programming, and scientific reasoning through a step-by-step "slow thinking" process [2, 28]. However, many real-world tasks necessitate the use of external tools for their completion. While some studies have explored new paradigms for integrating tool use within the reasoning process, such as Search-o1 [25], DeepResearcher [74], and ToRL [27], these approaches are often restricted to a limited set of predefined tools, such as web search, page browsing, and coding (Figure 2(b)). This constrained set of tools significantly hinders their applicability to a wide range of complex, real-world scenarios.

To address these challenges, we introduce **DeepAgent**, an end-to-end deep reasoning agent that can complete an entire task by dynamically retrieving and calling tools within a single, coherent agentic reasoning process. As depicted in Figure 2(c), DeepAgent operates by autonomously thinking, searching for tools, and executing actions. This paradigm shifts away from traditional, predefined workflows that rely on predefined tools, task planning, and iterative tool use, where each generation step focuses only on the immediate objective. Instead, DeepAgent maintains a global perspective on

* Work done during internship at Xiaohongshu.

Conference'17, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/xxxxxxx.xxxxxxx

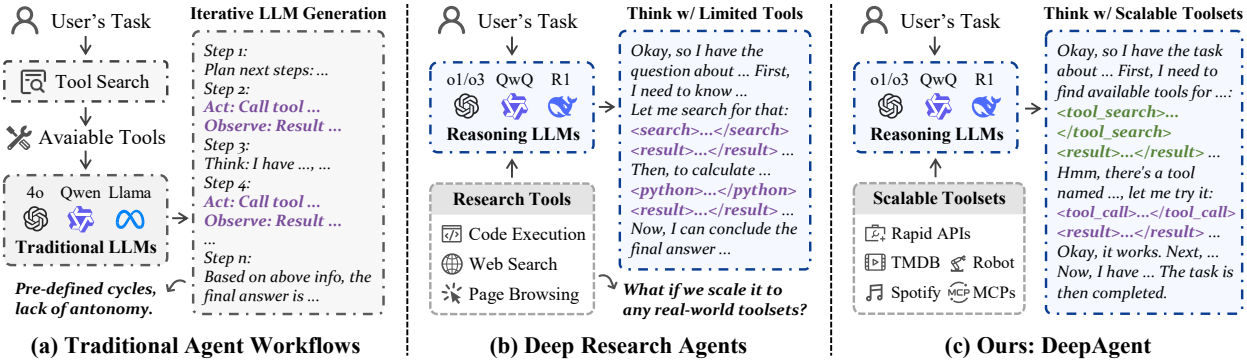


Figure 2: Comparison of agent paradigms: (a) Traditional agents with predefined workflows, (b) Deep Research agents that can autonomously call limited tools, and (c) Our DeepAgent, a fully autonomous reasoning agent that dynamically discovers and invokes helpful tools, all within a continuous agentic reasoning process.

the entire task, unconstrained by the need to deliberate on specific, isolated operations. Tools are not pre-retrieved in advance but are dynamically discovered on an as-needed basis, thereby fully unlocking the autonomous potential of the large reasoning model.

To empower DeepAgent to thoroughly and robustly explore new tools and navigate complex environments during long-horizon interactions, we equip it with memory management capabilities. We introduce an **Autonomous Memory Folding** strategy that allows DeepAgent to consolidate its previous thoughts and interaction history into a *structured memory schema* at any point during its thinking before resuming the agentic reasoning process. This mechanism not only saves tokens and enhances reasoning efficiency over extended interactions but also provides the agent an opportunity to "take a breath", preventing it from becoming trapped in wrong exploration paths and enabling it to reconsider its strategy, thus improving the overall success rate. To mitigate information loss during this folding process, we introduce a *brain-inspired memory architecture* comprising episodic memory, working memory, and tool memory, all structured with an agent-usable data schema to ensure the stability and utility of the compressed memory.

To enhance DeepAgent's proficiency in mastering these mechanisms, we propose **ToolPO**, an end-to-end reinforcement learning (RL) training method tailored for general tool use. Existing agentic RL training in general domains presents two significant challenges: (1) The reliance on a multitude of real-world APIs during training can lead to instability, slow execution, and high costs. To prevent this, we leverage *LLM-simulated APIs*, which enhance the stability and efficiency of the training process. (2) A sparse reward based solely on the final outcome is often insufficient to guarantee the accuracy of intermediate tool calls. We address this by implementing *tool-call advantage attribution*, which precisely assigns credit to the specific tokens responsible for correct tool invocations, thereby providing a more granular and effective learning signal.

We conduct extensive experiments on a wide range of benchmarks. For **(1) General Tool-Use Tasks**, we evaluate DeepAgent on ToolBench, API-Bank, TMDB, Spotify, and ToolHop, which feature toolsets scaling from tens to over ten thousand distinct tools. For **(2) Downstream Applications**, we test its performance on ALFWorld, WebShop, GAIA, and Humanity's Last Exam (HLE), which require the use of domain-specific toolsets. The overall results in Figure 1 show that DeepAgent achieves superior performance across all scenarios.

Our main contributions are summarized as follows:

(1) We propose DeepAgent, the first agentic framework that enables reasoning models to autonomously think, discover tools, and execute actions within a unified reasoning process, empowering LRMs to harness toolsets of arbitrary scale and generalize to complex real-world tasks.
(2) We introduce an autonomous memory folding mechanism, complemented by a brain-inspired memory design. This endows the agent with the ability to "take a breath" and reconsider its exploration strategies following unsuccessful attempts.
(3) We propose an end-to-end reinforcement learning training methodology for general-purpose tool use, ensuring stability and efficiency in large-scale tool execution during training, as well as accuracy in tool invocation during reasoning.
(4) We conduct extensive experiments across eight benchmarks, demonstrating DeepAgent's superior tool-use capabilities and high adaptability to real-world tasks.

2 Related Work

2.1 Large Reasoning Models

Large Reasoning Models (LRMs) [5, 16] have demonstrated significant performance improvements in mathematical, scientific, and coding tasks by employing step-by-step slow thinking processes before generating final responses. Existing research has explored various approaches to elicit extended Chain-of-Thought (CoT) reasoning [58] from models, including data synthesis for Supervised Fine-Tuning (SFT) [33, 36, 69], and end-to-end RL [5, 14]. Additionally, substantial work has investigated optimization strategies for reasoning models, such as advanced RL training algorithms [70, 73] and improving reasoning efficiency [3, 65]. However, models relying solely on parametric knowledge face inherent limitations and cannot interact with the real world. Recent studies have begun exploring tool-augmented reasoning approaches, including Search-o1 [25], Search-R1 [18], ToRL [27], DeepResearcher [74], and SimpleTIR [63]. However, these methods typically support only

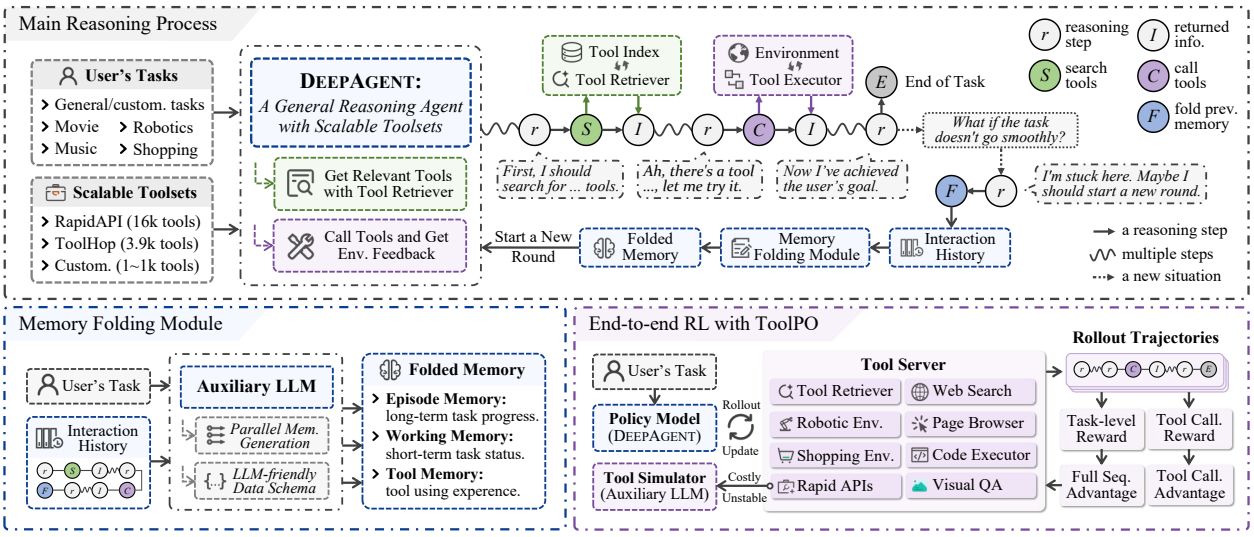


Figure 3: Overview of the DeepAgent framework. The main reasoning model autonomously discovers tools, executes actions, and folds previous memory to restart with structured memories, all within a unified thinking process. The DeepAgent is trained end-to-end with ToolPO, an RL method that uses a tool simulator to simulate large-scale real-world tool APIs, and rewards both final task success and correct intermediate tool calls through fine-grained advantage attribution.

a limited set of research-oriented tools, such as web search, page browsing, and code execution, which constrains their applicability to real-world scenarios that demand access to more diverse tools.

2.2 Autonomous Agents

LLM-powered autonomous agents accomplish real-world tasks by invoking external tools to interact with their environment [7, 15, 21, 23, 30, 38, 41, 52, 53, 59, 72]. Current agent methodologies, including ReAct [67], Plan-and-Solve [54], Reflextion [45], and CodeAct [56], predominantly follow predefined workflows with fixed execution patterns. This rigid structure limits their ability to fully leverage the autonomous decision-making and deep reasoning capabilities of advanced reasoning models. Recent efforts have investigated training LLMs to autonomously invoke tools through data synthesis and SFT methods [9, 48, 62] and RL training frameworks [4, 6, 8, 10, 11, 17, 22, 29, 31, 49, 57, 60]. However, most existing methods rely on pre-selected, labeled tools, which limit their applicability to real-world scenarios. Real-world tasks are highly variable and require access to diverse toolsets that cannot be predetermined, aligning with the emerging Model Context Protocol (MCP) [13] paradigm. Although some prior work has explored tool retrieval mechanisms [37, 43, 55], most approaches conduct only a single upfront retrieval step and incorporate the retrieved tools, with limited exploration of dynamic tool discovery during task execution. Therefore, we aim to develop a deep reasoning agent capable of dynamically discovering and invoking helpful tools from scalable toolsets to address more generalized real-world tasks.

3 Methodology

In this section, we first formulate the task of autonomous agentic reasoning. Then, we provide a detailed overview of the DeepAgent framework. Finally, we elaborate on the core components of DeepAgent, including the mechanism for autonomous tool use and memory folding, the brain-inspired memory schema, and our end-to-end reinforcement learning training method, ToolPO.

3.1 Problem Formulation

We frame the agent's task as a sequential decision-making process. The agent receives a user-provided question Q and an instruction I, and interacts with an environment over a series of steps $t = 1, \ldots, T$ to accomplish the specified goal. The environment provides access to a collection of tools $\mathcal{T}$ at an arbitrary scale.

At each step t, the agent's state s_t consists of the history of all previous actions and their resulting observations, i.e., $s_t = (a_1, o_1, \ldots, a_{t-1}, o_{t-1})$. The agent, driven by a policy π parameterized by θ, selects an action a_t based on the current state, the user question, and the instruction:

$$a_t \sim \pi_\theta(\cdot | s_t, Q, I). \tag{1}$$

An action a_t can be one of four types:

- **Internal Thought (a_t^{think})**: A textual reasoning step generated by the LRM to analyze the problem or plan its next steps. The corresponding observation o_t is typically empty.
- **Tool Search (a_t^{search})**: A natural language query q_s to find relevant tools from the toolset $\mathcal{T}$. The observation o_t is a list of retrieved tools.
- **Tool Call (a_t^{call})**: The invocation of a specific tool $\tau \in \mathcal{T}$ with a set of arguments. The observation o_t is the execution result returned by the tool.
- **Memory Fold (a_t^{fold})**: A special action to compress the interaction history s_t into a structured memory summary. The subsequent state s_{t+1} is then initialized with this compressed memory.

The sequence of states, actions, and observations forms a trajectory $\tau = (s_1, a_1, o_1, \ldots, s_T, a_T, o_T)$. The process terminates when the

agent completes the task or reaches a maximum step limit. The objective is to learn an optimal policy π_θ^* that maximizes the expected cumulative reward for a given task:

$$\pi_\theta^* = \arg\max_{\pi_\theta} \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)], \tag{2}$$

where $R(\tau)$ is a reward function that evaluates the overall success of the trajectory τ.

3.2 Overview of the DeepAgent Framework

As illustrated in Figure 3, the DeepAgent framework is architected around a main reasoning process, which is supported by several auxiliary mechanisms to ensure robustness and efficiency.

- **Main Reasoning Process**: The core of DeepAgent is a powerful large reasoning model that drives the entire task-completion process. In a single stream of thought, the LRM autonomously reasons about the task, dynamically discovers necessary tools, executes actions, and manages its own memory. This unified approach departs from traditional, rigid agent workflows, allowing the LRM to maintain a global perspective on the task.
- **Auxiliary Mechanisms**: DeepAgent employs an auxiliary LLM to handle complex interactions with large toolsets and manage long histories. This background model enhances system stability by: (1) filtering and summarizing retrieved tool documentation if it's too lengthy, (2) denoising and condensing verbose information returned from tool calls, and (3) compressing long interaction histories into a structured memory. This division of labor allows the main LRM to concentrate on high-level strategic reasoning.

3.3 Autonomous Tool Search and Calling

DeepAgent's main LRM performs all actions by generating specific textual prompts within its continuous reasoning process. These actions are then intercepted and executed by the system.

Tool Search. When the agent determines it needs a tool, it generates a tool search query q_s encapsulated within special tokens: `<tool_search>` q_s `</tool_search>`. The system's tool retriever operates via dense retrieval. First, we build an index by pre-computing an embedding $E(d_i)$ for the documentation d_i of each tool $\tau_i \in \mathcal{T}$ using an embedding model E. During inference, given the query q_s, the system retrieves the top-k tools by ranking them based on the cosine similarity $\text{sim}(\cdot,\cdot)$:

$$\mathcal{T}_{\text{retrieved}} = \underset{\tau_i\in\mathcal{T}}{\text{top-k}}\left(\text{sim}(E(q_s), E(d_i))\right). \tag{3}$$

The retrieved tool documentation is then processed by the auxiliary LLM —summarized if too lengthy, otherwise provided directly— and returned to the main LRM's context: `<tool_search_result>` relevant tools `</tool_search_result>`.

Tool Call. To execute a tool, the agent generates a structured call including the tool's name and arguments: `<tool_call>` {"name": "tool_name", "arguments": ...} `</tool_call>`. The framework parses this call, executes the tool, and captures the output. This output is, if necessary, summarized by the auxiliary LLM to ensure it is concise and helpful, before being fed back into the reasoning context: `<tool_call_result>` helpful information `</tool_call_result>`.

3.4 Autonomous Memory Folding and Brain-Inspired Memory Schema

The agent can trigger memory folding at any logical point in its reasoning process—such as after completing a sub-task or realizing an exploration path was incorrect—by generating a special token: `<fold_thought>`. Upon detecting this token, the system initiates the memory folding process. The auxiliary LLM (parameterized by θ_{aux}) processes the entire preceding interaction history s_t and generates three structured memory components in parallel:

$$(M_E, M_W, M_T) = f_{\text{compress}}(s_t; \theta_{\text{aux}}). \tag{4}$$

These compressed episodic (M_E), working (M_W), and tool (M_T) memories then replace the raw interaction history, enabling the agent to proceed with a refreshed and condensed view of its progress while avoiding entrapment in incorrect exploration paths.

Inspired by human cognitive systems, the structured memory M_t is composed of three distinct components that are generated in parallel: $M_t = (M_E, M_W, M_T)$, where M_E, M_W, M_T denote episodic, working, and tool memories, respectively.

- **Episodic Memory (M_E)**: This component serves as a high-level log of the task, recording key events, major decision points, and sub-task completions. It provides the agent with long-term context regarding the overall task structure and its overarching goals.
- **Working Memory (M_W)**: This contains the most recent information, such as the current sub-goal, obstacles encountered, and near-term plans. It is the core component that ensures the continuity of the agent's reasoning across the memory fold.
- **Tool Memory (M_T)**: This consolidates all tool-related interactions, including which tools have been used, how they were invoked, and their effectiveness. It allows the agent to learn from its experiences, refining its tool selection and usage strategies.

To ensure that the compressed memory is stable and easily parsed by the agent, we employ an **agent-usable data schema** in JSON format instead of unstructured natural language. This structured format offers two main benefits: it maintains a controllable and predictable structure, and it mitigates the loss of critical details that can occur when summarizing long-form text. Details of the data schema are provided in Appendix D.

3.5 End-to-end RL Training with ToolPO

We train DeepAgent end-to-end with Tool Policy Optimization (ToolPO), an RL approach designed for general tool-using agents.

Training Data Collection. We first collect a diverse training dataset spanning four categories. To instill **general tool-use** capabilities, we use ToolBench [37]. For **real-world interaction**, we leverage ALFWorld [46] and WebShop [66]. To enhance **deep research** skills, we incorporate data from WebDancer [59] and WebShaperQA [50]. Lastly, to improve **mathematical reasoning** with code, we use DeepMath [12]. Further details are available in Appendix A.1.

Tool Simulator. Training an agent that interacts with thousands of real-world APIs is often impractical due to instability, latency, and cost. To address this, we develop an **LLM-based Tool Simulator**. This simulator, powered by an auxiliary LLM, mimics the responses of real-world APIs (e.g., RapidAPI). This approach provides a stable, efficient, and low-cost environment for robust RL training.

Table 1: Main results on general tool usage tasks, encompassing scenarios with both labeled tools and open-set tool retrieval over large-scale toolsets. We report Pass@1 metric for all tasks. For 32B models, the best results are in bold and the second are underlined. Results from larger or closed-sourced models are in gray color for reference.

Method	Backbone	ToolBench		API-Bank		TMDB		Spotify		ToolHop	
		Success	Path	Success	Path	Success	Path	Success	Path	Correct	Path
Scenario 1: Completing Tasks w/ Ground-truth Tools											
Workflow-based Methods											
ReAct	Qwen2.5-32B	41.0	64.7	60.4	68.3	46.0	65.3	29.8	56.3	37.6	49.1
CodeAct	Qwen2.5-32B	53.0	68.3	62.4	70.6	48.0	67.4	33.3	58.7	34.7	48.8
Plan-and-Solve	Qwen2.5-32B	52.0	65.4	58.4	67.5	51.0	71.6	28.1	54.8	39.2	49.7
ReAct	QwQ-32B	52.0	61.6	73.3	78.6	43.0	65.3	47.4	69.4	47.4	51.6
CodeAct	QwQ-32B	54.0	63.4	74.3	79.4	55.0	74.5	52.6	75.4	43.2	53.4
Plan-and-Solve	QwQ-32B	55.0	64.7	70.3	75.4	48.0	61.3	49.1	70.6	45.4	50.6
ReAct	Qwen2.5-72B	56.0	69.3	73.3	78.6	47.0	67.7	57.9	76.6	44.8	55.4
ReAct	GPT-4o	52.0	53.9	79.2	83.3	77.0	89.3	47.4	70.6	40.0	53.7
ReAct	DeepSeek-R1	57.0	68.3	71.3	76.2	76.0	89.0	64.9	81.3	50.2	61.8
Autonomous Tool Usage within Reasoning											
DeepAgent-32B-Base	QwQ-32B	63.0	74.3	**76.2**	**81.0**	85.0	92.0	70.2	89.3	49.1	59.8
DeepAgent-32B-RL	QwQ-32B	**69.0**	**78.6**	75.3	80.2	**89.0**	**94.8**	**75.4**	**92.0**	**51.3**	**62.5**
Scenario 2: Completing Tasks w/ Open-Set Tool Retrieval											
Workflow-based Methods											
ReAct	Qwen2.5-32B	55.0	20.8	16.0	42.0	11.0	34.5	7.0	25.4	13.2	17.9
CodeAct	Qwen2.5-32B	51.0	19.0	22.0	49.6	19.0	46.8	10.5	31.6	12.7	17.4
Plan-and-Solve	Qwen2.5-32B	54.0	20.4	18.0	42.8	15.0	40.5	8.8	26.3	12.0	16.3
ReAct	QwQ-32B	44.0	19.0	20.0	52.7	18.0	40.3	22.8	45.5	27.1	22.3
CodeAct	QwQ-32B	48.0	21.6	16.0	45.0	31.0	52.8	24.6	49.6	29.0	26.1
Plan-and-Solve	QwQ-32B	45.0	19.6	18.0	44.3	24.0	46.8	19.3	42.7	25.7	20.8
ReAct	Qwen2.5-72B	52.0	21.6	14.0	38.9	28.0	50.7	21.1	48.5	21.1	19.9
ReAct	GPT-4o	41.0	28.9	18.0	42.8	35.0	56.8	17.5	26.3	24.1	28.6
ReAct	DeepSeek-R1	47.0	22.3	12.0	57.3	34.0	53.1	29.8	51.7	36.2	32.9
Autonomous Tool Retrieval and Usage within Reasoning											
DeepAgent-32B-Base	QwQ-32B	60.0	35.7	22.0	61.8	52.0	71.8	49.1	68.6	38.4	40.3
DeepAgent-32B-RL	QwQ-32B	**64.0**	**37.2**	**24.0**	**64.9**	**55.0**	**74.3**	**50.9**	**74.4**	**40.6**	**40.5**

Global and Tool-Call Advantage Attribution. For each input prompt, we sample a group of K trajectories $\{\tau_1, \ldots, \tau_K\}$. ToolPO defines two distinct reward components. The first is a reward for overall task success, $R_{\text{succ}}(\tau)$, which is a task success score reflecting the quality of the final outcome (e.g., the accuracy of the final answer). The second is a tool-call reward, $R_{\text{action}}(\tau)$, which reflects the quality of intermediate actions. This action-level reward is composed of rewards for correct tool invocations and efficient memory folding. Specifically, $R_{\text{action}}(\tau) = \lambda_1 \sum_{t=1}^{T} C(a_t^{\text{call}}) + \lambda_2 S_{\text{pref}}(\tau)$, where $C(a_t^{\text{call}})$ is 1 if a tool call is correct and 0 otherwise. $S_{\text{pref}}(\tau)$ is a preference score encouraging efficient use of memory folding, defined by comparing a trajectory with folding (τ_{fold}) to one without (τ_{direct}): $S_{\text{pref}} = (L(\tau_{\text{direct}}) - L(\tau_{\text{fold}}))/(L(\tau_{\text{direct}}) + L(\tau_{\text{fold}}))$.

Based on these rewards, we compute two separate group-relative advantages. The task success advantage for trajectory τ_k is:

$$A_{\text{succ}}(\tau_k) = R_{\text{succ}}(\tau_k) - \frac{1}{K} \sum_{j=1}^{K} R_{\text{succ}}(\tau_j). \tag{5}$$

This advantage is attributed to all generated tokens in the trajectory, providing a global learning signal. Similarly, the action-level advantage is:

$$A_{\text{action}}(\tau_k) = R_{\text{action}}(\tau_k) - \frac{1}{K} \sum_{j=1}^{K} R_{\text{action}}(\tau_j). \tag{6}$$

Crucially, this advantage is attributed *only* to the specific tokens that constitute the tool call and memory folding actions. This fine-grained credit assignment provides a more targeted signal for learning correct and efficient tool use.

Optimization Objective. The total advantage for a given token y_i in trajectory τ_k is the sum of the global and local advantages:

$$A(y_i) = A_{\text{succ}}(\tau_k) + M(y_i) \cdot A_{\text{action}}(\tau_k), \tag{7}$$

where $M(y_i)$ is a mask that is 1 if y_i is part of a tool-call or memory-fold token sequence, and 0 otherwise. ToolPO then optimizes the policy using a clipped surrogate objective function:

$$\mathcal{L}_{\text{ToolPO}}(\theta) = \mathbb{E}_{\tau_k}\left[\sum_{i=1}^{|\tau_k|} \min\left(\rho_i(\theta)A(y_i), \text{clip}(\rho_i(\theta), 1-\epsilon, 1+\epsilon)A(y_i)\right)\right], \tag{8}$$

Here, $\rho_i(\theta) = \frac{\pi_\theta(y_i|y_{<i},s)}{\pi_{\theta_{\text{old}}}(y_i|y_{<i},s)}$ is the probability ratio for token y_i. This objective encourages the model to increase the probability of both intermediate actions and end-to-end task accomplishment that exhibit positive relative advantage, thereby ensuring stable and effective policy updates.

Table 2: Main results on downstream task applications, spanning Embodied AI (ALFWorld), Online Shopping (WebShop), General AI Assistants (GAIA), and Humanity's Last Exam (HLE). We report Pass@1 for all tasks. For 32B models, the best results are in bold and the second are underlined. Results from larger or closed-sourced models are in gray color for reference.

Method	Backbone	ALFWorld		WebShop		GAIA				HLE		
		Success	Path	Success	Score	Text	MM	File	All	Text	MM	All
Completing Tasks w/ Task-specific Toolsets												
Workflow-based Methods												
ReAct	Qwen2.5-32B	60.4	79.1	6.0	28.8	25.2	16.7	13.2	21.2	6.5	7.1	6.6
CodeAct	Qwen2.5-32B	65.7	83.3	12.4	34.5	28.2	20.8	18.4	24.8	7.5	8.0	7.6
Reflextion	Qwen2.5-32B	66.4	86.0	9.2	31.6	29.1	20.8	18.4	25.5	5.9	5.3	5.8
Plan-and-Solve	Qwen2.5-32B	63.4	80.4	7.6	29.3	27.2	16.7	15.8	23.0	7.2	6.2	7.0
ReAct	QwQ-32B	82.1	87.8	17.2	45.3	35.0	8.3	36.8	31.5	13.2	8.8	12.2
CodeAct	QwQ-32B	78.4	86.2	18.0	46.4	38.8	20.8	31.6	34.5	14.2	8.0	12.8
Reflextion	QwQ-32B	85.1	88.4	21.6	50.4	37.9	20.8	36.8	35.2	11.9	7.1	10.8
Plan-and-Solve	QwQ-32B	79.1	84.7	16.0	43.8	36.9	16.7	34.2	33.3	12.9	9.7	12.2
AgentLM*	Llama2-70B	86.0	-	-	64.9	-	-	-	-	-	-	-
ReAct	Qwen2.5-72B	86.5	86.5	22.0	44.5	32.0	20.8	31.6	30.3	9.0	8.0	8.8
ReAct	DeepSeek-R1	79.1	85.8	19.6	49.7	43.7	29.2	39.5	40.6	14.2	8.8	13.0
ReAct	GPT-4o	65.7	87.8	15.6	52.5	35.0	16.7	36.8	32.7	13.2	10.6	12.6
ReAct	Claude-4	93.3	91.5	20.4	56.6	56.3	37.5	52.6	52.7	15.5	16.8	15.8
Autonomous Tool Usage within Reasoning												
Deep Research	OpenAI (o3)	-	-	-	-	-	-	-	67.4	-	-	26.6
WebThinker	QwQ-32B	-	-	-	-	48.5	25.0	13.2	37.0	14.2	8.8	13.0
HiRA	QwQ-32B	84.3	87.6	23.2	51.9	44.7	33.3	42.1	42.5	14.5	10.6	13.6
DeepAgent-32B-Base	QwQ-32B	88.1	91.4	32.0	55.4	49.5	**37.5**	44.7	46.7	19.1	13.3	17.8
DeepAgent-32B-RL	QwQ-32B	**91.8**	**92.0**	**34.4**	**56.3**	**58.3**	33.3	**52.6**	**53.3**	**21.7**	**15.0**	**20.2**

4 Experimental Settings

4.1 Tasks and Datasets

We conduct extensive experiments on a wide range of benchmarks, including general tool-use and downstream applications.

General Tool-Use. These benchmarks encompass a broad range of distinct tools, scaling from tens to over ten thousand, making them ideal for evaluating the scalability of different approaches. We utilize four representative scenarios: **ToolBench [37]**, based on over 16,000 real-world APIs, for which we use the G3 subset requiring multi-step, multi-tool calls; **API-Bank [24]**, which includes 314 human-annotated dialogues with 73 APIs and 753 API calls, to assess planning, retrieval, and calling capabilities; **RestBench [47]**, comprising scenarios from the TMDB movie database (54 tools, avg. 2.3 calls/question) and the Spotify music player (40 tools, avg. 2.6 calls/question) to simulate typical REST applications; and **ToolHop [68]**, a multi-hop reasoning dataset with 3,912 locally executable tools that necessitate 3 to 7 sequential tool calls per task. For these tasks, we adopt two settings: given ground-truth tools and given entire toolsets with tool retrieval capabilities.

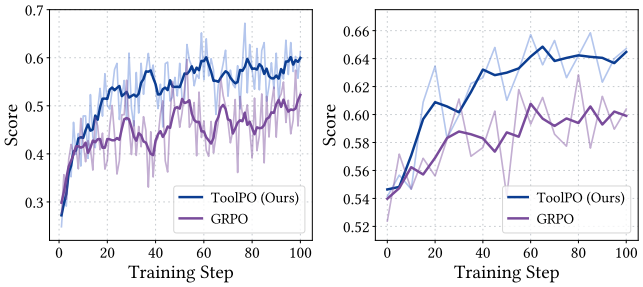
Downstream Applications. We evaluate our approach on several downstream applications that require domain-specific toolsets. These include **ALFWorld [46]**, a text-based embodied AI task where agents complete goals using nine basic actions (e.g., move, take); **WebShop [66]**, an online shopping environment with 'search' and 'click' actions to fulfill users' specific product purchasing requirements; **GAIA [32]**, a complex information-seeking benchmark where we equip the agent with tools for web search, page browsing, Visual Question Answering (VQA), code compilation, and file reading; and **Humanity's Last Exam (HLE) [35]**, a set of highly difficult reasoning problems, for which we provide code, search, page browsing, and VQA tools. These benchmarks test the agent's ability to perform long-horizon planning and robust interaction in complex, real-world scenarios. For this category of tasks, we provide agents with task-specific toolsets.

4.2 Baselines

Our baselines include: (1) **Workflow-based Methods**: ReAct [67] alternates explicit reasoning with environment actions in a Reason-Act-Observe loop. CodeAct [56] expresses actions as executable Python code that runs in an interpreter. Plan-and-Solve [54] first sketches a high-level plan and then executes it step by step. Reflexion [44] enhances learning through verbal self-reflection after failed attempts. AgentLM [71] uses instruction tuning to enhance general agent capabilities of LLMs. (2) **Autonomous Tool Usage within Reasoning**: WebThinker [26] interleaves thinking with web search and deep web exploration. HiRA [20] introduces a hierarchical agent architecture where a meta planner decomposes tasks, a coordinator routes subtasks, and specialized executors solve them with dual-channel memory. OpenAI Deep Research [34] is an agentic system based on reasoning models.

4.3 Implementation Details

We use QwQ-32B [51] as DeepAgent's backbone model, with Qwen2.5-32B-Instruct [40] as the auxiliary model in our main results. Text generation employs a maximum of 81,920 tokens with temperature 0.7, top_p 0.8, top_k 20, and repetition penalty 1.05. Web search



(a) Reward Scores (b) Validation Scores

Figure 4: Visualization of training dynamics, including (a) reward scores and (b) validation scores across training steps.

Table 3: Ablation studies on the components of DeepAgent, where the best results are in bold.

Method	Tool-Usage		Application		Avg.
	ToolB.	ToolH.	WebS.	GAIA	
DeepAgent-32B-RL	**64.0**	**40.6**	**34.4**	**53.3**	**48.1**
w/o Training (Base)	60.0	38.4	32.0	46.7	44.3
w/o Memory Folding	63.0	36.6	32.4	44.7	44.2
w/o Tool Simulation	62.0	35.2	33.6	48.5	44.8
w/o Tool Adv. Attribution	62.0	39.6	33.2	49.5	46.1

and page browsing are implemented using Google Serper API and Jina Reader API, respectively. The VQA tool is based on Qwen2.5-VL-32B-Instruct [1]. Tool retrieval is performed using bge-large-en-v1.5 [61]. Training consists of 100 steps of ToolPO with batch size 64, $\lambda_1 = \lambda_2 = 1$, rollout size $K = 8$, and maximum sequence length 32,768. Additional details are provided in Appendix C. All experiments are conducted on 64 NVIDIA H20-141GB GPUs.

5 Experimental Results

5.1 Main Results on General Tool Usage Tasks

Table 1 presents the results on general tool usage, leading to several key observations. **(1) DeepAgent's End-to-End Reasoning Surpasses Workflow-Based Methods.** DeepAgent's holistic agentic process consistently outperforms rigid, predefined workflows. For instance, on labeled-tool tasks, DeepAgent-32B-RL achieves success rates of 89.0% on TMDB and 75.4% on Spotify, substantially exceeding the strongest 32B baseline scores of 55.0% and 52.6%, respectively. This underscores the benefit of a holistic agentic process over rigid, predefined action cycles. **(2) DeepAgent Maintains Robustness in Open-Set Scenarios.** This advantage is more pronounced in open-set scenarios where dynamic tool discovery is critical. On ToolBench and ToolHop, DeepAgent-32B-RL achieves success rates of 64.0% and 40.6%, respectively, far exceeding the top baseline scores of 54.0% and 29.0%. This demonstrates that DeepAgent's strategy of dynamically discovering tools as needed within the reasoning process is far more robust and scalable in realistic open-set scenarios. **(3) ToolPO Training Further Improves Tool-Usage Capabilities.** The proposed ToolPO RL strategy provides significant further gains. The trained DeepAgent-32B-RL model consistently improves upon its base version, boosting success rates on ToolBench by up to 6.0% and on Spotify (labeled) by 5.2%. This validates the effectiveness of the ToolPO strategy, which uses an LLM-based tool simulator and fine-grained advantage attribution.

Table 4: Effectiveness analysis of autonomous tool retrieval strategy in open-set scenarios compared to pre-retrieved tool methods. Numbers in parentheses indicate toolset sizes.

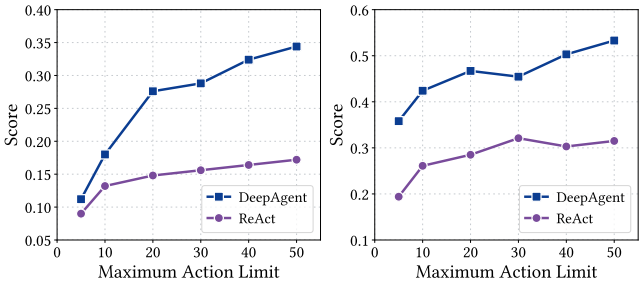
Method	ToolB. (16k)	ToolH. (3.9k)	TMDB (54)	Spotify (40)	Avg.
ReAct Workflow					
Input Retrieved Tool	35.0	25.4	14.0	15.0	22.4
Auto. Tool Retrieval	34.0	37.1	18.0	27.8	28.0
Plan-and-Solve Workflow					
Input Retrieved Tool	37.0	24.8	19.0	16.0	24.2
Auto. Tool Retrieval	45.0	25.7	24.0	19.3	28.5
End-to-end Agentic Reasoning (DeepAgent)					
Input Retrieved Tool	53.0	37.0	34.0	43.9	42.0
Auto. Tool Retrieval	**64.0**	**40.6**	**55.0**	**50.9**	**52.6**

5.2 Main Results on Downstream Applications

Table 2 shows the results on downstream applications, which require agents to handle long-horizon interactions in complex environments. **(1) The autonomous reasoning paradigm generally outperforms workflow-based methods.** On complex application tasks, methods that integrate tool usage into continuous reasoning consistently outperform rigid, predefined workflows. On GAIA, both DeepAgent-32B-Base (46.7) and HiRA (42.5) significantly exceed the best workflow-based method CodeAct (34.5). Similarly, on WebShop, DeepAgent-32B-Base (32.0) substantially surpasses CodeAct (18.0). This demonstrates that long-horizon interaction tasks require deep agentic reasoning capabilities to achieve superior task accomplishments. **(2) DeepAgent demonstrates superior performance across various application tasks.** DeepAgent achieves state-of-the-art performance among 32B models. On GAIA, DeepAgent-32B-RL scores 53.3 vs. HiRA's 42.5, and on ALFWorld reaches 91.8% vs. HiRA's 84.3%. This stems from DeepAgent's seamless integration of actions into coherent reasoning, enabling end-to-end execution with autonomous memory folding, which is advantages unavailable to workflow-constrained methods. **(3) ToolPO training further improves performance on downstream applications.** ToolPO training yields consistent gains over the base model. DeepAgent-32B-RL improves GAIA scores from 46.7 to 53.3 (+6.6) and ALFWorld success rates from 88.1% to 91.8% (+3.7), demonstrating that ToolPO effectively enhances reasoning and tool usage capabilities for complex task completion.

5.3 Analysis of Training Dynamics

Figure 4 shows the training dynamics of DeepAgent, including the reward scores and validation scores across training steps. As shown in the figure, **(1) DeepAgent trained with ToolPO achieves higher upper bounds on both reward and validation scores compared to the commonly used GRPO. (2) Moreover, the training reward exhibits less fluctuation than GRPO, demonstrating better training stability.** This indicates that using tool



(a) WebShop (b) GAIA

Figure 5: Scaling analysis of performance with respect to maximum action limits on WebShop and GAIA datasets.

simulators instead of directly training with unstable real-world APIs, along with employing tool-call process supervision, enables more stable and effective training of tool-usage capabilities.

5.4 Ablation Studies

We conduct ablation studies in Table 3 to validate the effectiveness of each component in DeepAgent. **(1) Importance of ToolPO Training:** Removing ToolPO training (the Base model) results in the most significant performance drop (from 48.1 to 44.3). This highlights the central role of our end-to-end RL method in enhancing tool use and complex task completion. **(2) Effectiveness of Memory Folding:** The absence of memory folding also leads to a substantial performance decline (average score drops to 44.2), particularly on the long-horizon task GAIA (from 53.3 to 44.7). This confirms that the autonomous memory folding mechanism, allowing the agent to "take a breath" and replan, is crucial for robust long-term interaction. **(3) Contribution of Training Strategies:** Removing the tool simulator and tool-call advantage attribution both lead to performance degradation. This validates that the tool simulator enables more stable training, and fine-grained advantage attribution provides precise learning signals.

5.5 Effectiveness of Tool Retrieval Strategies

To compare pre-retrieving tools versus autonomous discovery during task execution, we conduct experiments shown in Table 4. **(1) The on-demand nature of dynamic tool discovery yields superior performance and robust scalability.** Autonomous tool retrieval during reasoning consistently outperforms pre-retrieved tools across all frameworks, demonstrating the superiority of on-demand tool access in open-set scenarios. Performance gains are most pronounced on large toolsets like ToolBench (16k tools) and ToolHop (3.9k tools), indicating robust scalability for real-world tasks. **(2) DeepAgent synergizes better with dynamic retrieval.** Combined with autonomous tool retrieval, our framework achieves the best results by a large margin, scoring 52.6 on average versus 28.5 for the best workflow-based method. This demonstrates DeepAgent's architecture is uniquely suited for dynamic tool discovery.

5.6 Scaling Analysis of Action Limits

Figure 5 illustrates the performance of DeepAgent and ReAct on the WebShop and GAIA datasets as the maximum action limit is varied. The results yield several key insights. **(1) DeepAgent consistently and significantly outperforms the ReAct baseline across all tested action limits on both datasets**, demonstrating its superior effectiveness. **(2) For both agents, performance generally improves as the maximum number of actions increases.** This suggests that complex tasks benefit from a longer interaction horizon, allowing for more thorough exploration and reasoning. **(3) DeepAgent exhibits stronger scalability.** As the action limit increases, the performance gap between DeepAgent and ReAct widens, particularly on WebShop. This sustained gain suggests DeepAgent strategically selects effective, task-relevant actions, avoiding the wasteful steps that limit ReAct's scalability.

Table 5: Performance comparison with different reasoning model backbones, spanning MOE-based models with 30B and 235B parameters.

Method	Tool-Usage		Application			Avg.
	ToolB.	ToolH.	ALF.	WebS.	GAIA	
Qwen3-30B-A3B-Thinking						
ReAct	52.0	22.0	67.9	18.4	34.5	35.7
Plan-and-Solve	50.0	23.6	68.7	20.4	35.2	37.0
DeepAgent (Base)	**59.0**	**47.5**	**69.4**	**31.4**	**39.4**	**46.9**
Qwen3-235B-A22B-Thinking						
ReAct	61.0	40.9	79.9	21.6	36.4	45.1
Plan-and-Solve	63.0	43.0	78.4	24.4	38.4	46.0
DeepAgent (Base)	**67.0**	**48.2**	**85.8**	**37.2**	**51.5**	**55.7**

5.7 Generalization Across Different Backbones

Table 5 shows the performance of DeepAgent with different backbone large reasoning models, including Qwen3-30B-A3B-Thinking and Qwen3-235B-A22B-Thinking [64]. **(1) DeepAgent consistently outperforms workflow-based methods.** With both the 30B and 235B MoE-based reasoning models as backbones, DeepAgent maintains a significant performance margin over ReAct and Plan-and-Solve, demonstrating the generalizability of its agentic reasoning approach. **(2) DeepAgent scales effectively with larger models.** While all methods benefit from scaling the backbone from a 30B to a 235B model, DeepAgent shows the largest absolute performance gains on complex application tasks.

6 Conclusion

In this work, we introduce DeepAgent, an end-to-end reasoning agent that unifies thinking, tool discovery, and execution into a single, coherent agentic reasoning process. To enable robust long-horizon interaction, we propose an autonomous memory folding mechanism that compresses interaction history into a structured memory, allowing the agent to "take a breath" and reconsider its strategy. We also introduce ToolPO, an end-to-end RL method that leverages LLM simulated APIs for stable training and fine-grained advantage attribution for precise credit assignment to tool invocations. Extensive experiments on general tool-use and downstream applications demonstrate that DeepAgent significantly outperforms various baseline agents, particularly in open-set scenarios requiring dynamic tool discovery over scalable toolsets.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. 2025. Qwen2.5-VL Technical Report. *CoRR* abs/2502.13923 (2025). arXiv:2502.13923 doi:10.48550/ARXIV.2502.13923

[2] Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards Reasoning Era: A Survey of Long Chain-of-Thought for Reasoning Large Language Models. *CoRR* abs/2503.09567 (2025). arXiv:2503.09567 doi:10.48550/ARXIV.2503.09567

[3] Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2024. Do NOT Think That Much for 2+3=? On the Overthinking of o1-Like LLMs. arXiv:2412.21187 [cs.CL] https://arxiv.org/abs/2412.21187

[4] Yifei Chen, Guanting Dong, and Zhicheng Dou. 2025. Toward Effective Tool-Integrated Reasoning via Self-Evolved Preference Learning. arXiv:2509.23285 [cs.AI] https://arxiv.org/abs/2509.23285

[5] 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, and S. S. Li. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *CoRR* abs/2501.12948 (2025). arXiv:2501.12948 doi:10.48550/ARXIV.2501.12948

[6] Guanting Dong, Licheng Bao, Zhongyuan Wang, Kangzhi Zhao, Xiaoxi Li, Jiajie Jin, Jinghan Yang, Hangyu Mao, Fuzheng Zhang, Kun Gai, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2025. Agentic Entropy-Balanced Policy Optimization. arXiv:2510.14545 [cs.LG] https://arxiv.org/abs/2510.14545

[7] Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui Zhou, Zhicheng Dou, and Ji-Rong Wen. 2025. Tool-Star: Empowering LLM-Brained Multi-Tool Reasoner via Reinforcement Learning. *CoRR* abs/2505.16410 (2025). arXiv:2505.16410 doi:10.48550/ARXIV.2505.16410

[8] Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2025. Agentic Reinforced Policy Optimization. *CoRR* abs/2507.19849 (2025). arXiv:2507.19849 doi:10.48550/ARXIV.2507.19849

[9] Runnan Fang, Shihao Cai, Baixuan Li, Jialong Wu, Guangyu Li, Wenbiao Yin, Xinyu Wang, Xiaobin Wang, Liangcai Su, Zhen Zhang, Shibin Wu, Zhengwei Tao, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. Towards General Agentic Intelligence via Environment Scaling. arXiv:2509.13311 [cs.CL] https://arxiv.org/abs/2509.13311

[10] Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. 2025. ReTool: Reinforcement Learning for Strategic Tool Use in LLMs. arXiv:2504.11536 [cs.CL] https://arxiv.org/abs/2504.11536

[11] Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. 2025. Beyond Ten Turns: Unlocking Long-Horizon Agentic Search with Large-Scale Asynchronous RL. *CoRR* abs/2508.07976 (2025). arXiv:2508.07976 doi:10.48550/ARXIV.2508.07976

[12] Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. DeepMath-103K: A Large-Scale, Challenging, Decontaminated, and Verifiable Mathematical Dataset for Advancing Reasoning. (2025). arXiv:2504.11456 [cs.CL] https://arxiv.org/abs/2504.11456

[13] Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. 2025. Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions. *CoRR* abs/2503.23278 (2025). arXiv:2503.23278 doi:10.48550/ARXIV.2503.23278

[14] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model. *CoRR* abs/2503.24290 (2025). arXiv:2503.24290 doi:10.48550/ARXIV.2503.24290

[15] Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, Jianye Hao, Kun Shao, and Jun Wang. 2025. Deep Research Agents: A Systematic Examination And Roadmap. *CoRR* abs/2506.18096 (2025). arXiv:2506.18096 doi:10.48550/ARXIV.2506.18096

[16] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. OpenAI o1 System Card. *arXiv preprint arXiv:2412.16720* (2024).

[17] Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, Tianyu Pang, and Wenhu Chen. 2025. VerlTool: Towards Holistic Agentic Reinforcement Learning with Tool Use. arXiv:2509.01055 [cs.AI] https://arxiv.org/abs/2509.01055

[18] Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning. *CoRR* abs/2503.09516 (2025). arXiv:2503.09516 doi:10.48550/ARXIV.2503.09516

[19] Haolin Jin, Linghan Huang, Haipeng Cai, Jun Yan, Bo Li, and Huaming Chen. 2024. From LLMs to LLM-based Agents for Software Engineering: A Survey of Current, Challenges and Future. *CoRR* abs/2408.02479 (2024). arXiv:2408.02479 doi:10.48550/ARXIV.2408.02479

[20] Jiajie Jin, Xiaoxi Li, Guanting Dong, Yuyao Zhang, Yutao Zhu, Zhao Yang, Hongjin Qian, and Zhicheng Dou. 2025. Decoupled Planning and Execution: A Hierarchical Reasoning Framework for Deep Search. *CoRR* abs/2507.02652 (2025). arXiv:2507.02652 doi:10.48550/ARXIV.2507.02652

[21] Jiajie Jin, Yuyao Zhang, Yimeng Xu, Hongjin Qian, Yutao Zhu, and Zhicheng Dou. 2025. FinSight: Towards Real-World Financial Deep Research. arXiv:2510.16844 [cs.CL] https://arxiv.org/abs/2510.16844

[22] Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A. Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. 2025. ACON: Optimizing Context Compression for Long-horizon LLM Agents. arXiv:2510.00615 [cs.AI] https://arxiv.org/abs/2510.00615

[23] Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Dingchu Zhang, Xixi Wu, Jialong Wu, Xinyu Wang, Zile Qiao, Zhen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. WebSailor-V2: Bridging the Chasm to Proprietary Agents via Synthetic Data and Scalable Reinforcement Learning. *CoRR* abs/2509.13305 (2025). arXiv:2509.13305 doi:10.48550/ARXIV.2509.13305

[24] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, 3102–3116. doi:10.18653/V1/2023.EMNLP-MAIN.187

[25] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic Search-Enhanced Large Reasoning Models. *CoRR* abs/2501.05366 (2025). arXiv:2501.05366 doi:10.48550/ARXIV.2501.05366

[26] Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yutao Zhu, Yongkang Wu, Ji-Rong Wen, and Zhicheng Dou. 2025. WebThinker: Empowering Large Reasoning Models with Deep Research Capability. *CoRR* abs/2504.21776 (2025). arXiv:2504.21776 doi:10.48550/ARXIV.2504.21776

[27] Xuefeng Li, Haoyang Zou, and Pengfei Liu. 2025. ToRL: Scaling Tool-Integrated RL. arXiv:2503.23383 [cs.CL] https://arxiv.org/abs/2503.23383

[28] Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhijiang Guo, Le Song, and Cheng-Lin Liu. 2025. From System 1 to System 2: A Survey of Reasoning Large Language Models. *CoRR* abs/2502.17419 (2025). arXiv:2502.17419 doi:10.48550/ARXIV.2502.17419

[29] Zhuofeng Li, Haoxiang Zhang, Seungju Han, Sheng Liu, Jianwen Xie, Yu Zhang, Yejin Choi, James Zou, and Pan Lu. 2025. In-the-Flow Agentic System Optimization for Effective Planning and Tool Use. arXiv:2510.05592 [cs.AI] https://arxiv.org/abs/2510.05592

[30] Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, Jiayuan Song, Zhengmao Zhu, Wenhu Chen, Pengyu Zhao, and Junxian He. 2025. WebExplorer: Explore and Evolve for Training Long-Horizon Web Agents. *CoRR* abs/2509.06501 (2025). arXiv:2509.06501 doi:10.48550/ARXIV.2509.06501

[31] Zichen Liu, Anya Sims, Keyu Duan, Changyu Chen, Simon Yu, Xiangxin Zhou, Haotian Xu, Shaopan Xiong, Bo Liu, Chenmien Tan, Chuen Yang Beh, Weixun Wang, Hao Zhu, Weiyan Shi, Diyi Yang, Michael Shieh, Yee Whye Teh, Wee Sun Lee, and Min Lin. 2025. GEM: A Gym for Agentic LLMs. arXiv:2510.01051 [cs.LG] https://arxiv.org/abs/2510.01051

[32] Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. GAIA: a benchmark for General AI Assistants. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net. https://openreview.net/forum?id=fibxvahvs3

[33] Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, et al. 2024. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *arXiv preprint arXiv:2412.09413* (2024).

[34] OpenAI. 2025. Introducing deep research. https://openai.com/index/introducing-deep-research.

[35] Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra, Adam Khoja, Ryan Kim, Jason Hausenloy, Oliver Zhang, Mantas Mazeika, Daron Anderson, Tung Nguyen, Mobeen Mahmood, Fiona Feng, Steven Y. Feng, Haoran Zhao, Michael Yu, Varun Gangal, Chelsea Zou, Zihan Wang, Jessica P. Wang, Pawan Kumar, Oleksandr Pokutnyi, Robert Gerbicz, Serguei Popov, John-Clark Levin, Mstyslav Kazakov, Johannes Schmitt, Geoff Galgon, Alvaro Sanchez, Yongki Lee, Will Yeadon, Scott Sauers, Marc Roth, Chidozie Agu, Søren Riis, Fabian Giska, Saiteja Utpala, Zachary Giboney, Gashaw M. Goshu, Joan of Arc Xavier, Sarah-Jane Crowson, Mohinder Maheshbhai Naiya, Noah Burns, Lennart Finke, Zerui Cheng, Hyunwoo Park, Francesco Fournier-Facio, John Wydallis, Mark Nandor, Ankit Singh, Tim Gehrunger, Jiaqi Cai, Ben McCarty, Darling Duclosel, Jungbae Nam, Jennifer Zampese, Ryan G. Hoerr, Aras Bacho, Gautier Abou Loume, Abdallah Galal, Hangrui Cao, Alexis C. Garretson, Damien Sileo, Qiuyu Ren, Doru Cojoc, Pavel Arkhipov, Usman Qazi, Lianghui Li, Sumeet Motwani, Christian Schröder de Witt, Edwin Taylor, Johannes Veith, Eric Singer, Taylor D. Hartman, Paolo Rissone, Jaehyeok Jin, Jack Wei Lun Shi, Chris G. Willcocks, Joshua Robinson, Aleksandar Mikov, Ameya Prabhu, Longke Tang, Xavier Alapont, Justine Leon Uro, Kevin Zhou, Emily de Oliveira Santos, Andrey Pupasov Maksimov, Edward Vendrow, Kengo Zenitani, Julien Guillod, Yuqi Li, Joshua Vendrow, Vladyslav Kuchkin, and Ng Ze-An. 2025. Humanity's Last Exam. *CoRR* abs/2501.14249 (2025). arXiv:2501.14249 doi:10.48550/ARXIV.2501.14249

[36] Yiwei Qin, Xuefeng Li, Haoyang Zou, Yixiu Liu, Shijie Xia, Zhen Huang, Yixin Ye, Weizhe Yuan, Hector Liu, Yuanzhi Li, et al. 2024. O1 Replication Journey: A Strategic Progress Report–Part 1. *arXiv preprint arXiv:2410.18982* (2024).

[37] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net. https://openreview.net/forum?id=dHng2O0Jjr

[38] Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. 2025. From Exploration to Mastery: Enabling LLMs to Master Tools via Self-Driven Interactions. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net. https://openreview.net/forum?id=QKBu1BOAwd

[39] Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. 2025. Tool learning with large language models: a survey. *Frontiers Comput. Sci.* 19, 8 (2025), 198343. doi:10.1007/S11704-024-40678-2

[40] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 Technical Report. arXiv:2412.15115 [cs.CL] https://arxiv.org/abs/2412.15115

[41] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. In *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*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html

[42] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. HybridFlow: A Flexible and Efficient RLHF Framework. *arXiv preprint arXiv: 2409.19256* (2024).

[43] Zhengliang Shi, Yuhan Wang, Lingyong Yan, Pengjie Ren, Shuaiqiang Wang, Dawei Yin, and Zhaochun Ren. 2025. Retrieval Models Aren't Tool-Savvy: Benchmarking Tool Retrieval for Large Language Models. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, 24497–24524. https://aclanthology.org/2025.findings-acl.1258/

[44] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *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*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html

[45] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2024).

[46] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. 2021. 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. https://openreview.net/forum?id=0IOX0YcCdTn

[47] Yifan Song, Weimin Xiong, Dawei Zhu, Cheng Li, Ke Wang, Ye Tian, and Sujian Li. 2023. RestGPT: Connecting Large Language Models with Real-World Applications via RESTful APIs. *CoRR* abs/2306.06624 (2023). arXiv:2306.06624 doi:10.48550/ARXIV.2306.06624

[48] Liangcai Su, Zhen Zhang, Guangyu Li, Zhuo Chen, Chenxi Wang, Maojia Song, Xinyu Wang, Kuan Li, Jialong Wu, Xuanzhong Chen, Zile Qiao, Zhongwang Zhang, Huifeng Yin, Shihao Cai, Runnan Fang, Zhengwei Tao, Wenbiao Yin, Chenxiong Qian, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. Scaling Agents via Continual Pre-training. arXiv:2509.13310 [cs.CL] https://arxiv.org/abs/2509.13310

[49] Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. 2025. Scaling Long-Horizon LLM Agent via Context-Folding. arXiv:2510.11967 [cs.CL] https://arxiv.org/abs/2510.11967

[50] Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. WebShaper: Agentically Data Synthesizing via Information-Seeking Formalization. *CoRR* abs/2507.15061 (2025). arXiv:2507.15061 doi:10.48550/ARXIV.2507.15061

[51] Qwen Team. 2024. Qwq: Reflect deeply on the boundaries of the unknown. *Hugging Face* (2024).

[52] Haoming Wang, Haoyang Zou, Huatong Song, Jiazhan Feng, Junjie Fang, Junting Lu, Longxiang Liu, Qinyu Luo, Shihao Liang, Shijue Huang, Wanjun Zhong, Yining Ye, Yujia Qin, Yuwen Xiong, Yuxin Song, Zhiyong Wu, Aoyan Li, Bo Li, Chen Dun, Chong Liu, Daoguang Zan, Fuxing Leng, Hanbin Wang, Hao Yu, Haobin Chen, Hongyi Guo, Jing Su, Jingjia Huang, Kai Shen, Kaiyu Shi, Lin Yan, Peiyao Zhao, Pengfei Liu, Qinghao Ye, Renjie Zheng, Shulin Xin, Wayne Xin Zhao, Wen Heng, Wenhao Huang, Wenqian Wang, Xiaobo Qin, Yi Lin, Youbin Wu, Zehui Chen, Zihao Wang, Baoquan Zhong, Xinchun Zhang, Xujing Li, Yuanfan Li, Zhongkai Zhao, Chengquan Jiang, Faming Wu, Haotian Zhou, Jinlin Pang, Li Han, Qi Liu, Qianli Ma, Siyao Liu, Songhua Cai, Wenqi Fu, Xin Liu, Yaohui Wang, Zhi Zhang, Bo Zhou, Guoliang Li, Jiajun Shi, Jiale Yang, Jie Tang, Li Li, Qihua Han, Taoran Lu, Woyu Lin, Xiaokang Tong, Xinyao Li, Yichi Zhang, Yu Miao, Zhengxuan Jiang, Zili Li, Ziyuan Zhao, Chenxin Li, Dehua Ma, Feng Lin, Ge Zhang, Haihua Yang, Hangyu Guo, Hongda Zhu, Jiaheng Liu, Junda Du, Kai Cai, Kuanye Li, Lichen Yuan, Meilan Han, Minchao Wang, Shuyue Guo, Tianhao Cheng, Xiaobo Ma, Xiaojun Xiao, Xiaolong Huang, Xinjie Chen, and Yidi Du. 2025. UI-TARS-2 Technical Report: Advancing GUI Agent with Multi-Turn Reinforcement Learning. *CoRR* abs/2509.02544 (2025). arXiv:2509.02544 doi:10.48550/ARXIV.2509.02544

[53] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on large language model based autonomous agents. *Frontiers Comput. Sci.* 18, 6 (2024), 186345. doi:10.1007/S11704-024-40231-1

[54] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, 2609–2634. doi:10.18653/V1/2023.ACL-LONG.147

[55] Renxi Wang, Xudong Han, Lei Ji, Shu Wang, Timothy Baldwin, and Haonan Li. 2025. ToolGen: Unified Tool Retrieval and Calling via Generation. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net. https://openreview.net/forum?id=XLMAMmowdY

[56] Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024. Executable Code Actions Elicit Better LLM Agents. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net. https://openreview.net/forum?id=jJ9BoXAfFa

[57] Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. 2025. RAGEN: Understanding Self-Evolution in LLM Agents via Multi-Turn Reinforcement Learning. *CoRR* abs/2504.20073 (2025). arXiv:2504.20073 doi:10.48550/ARXIV.2504.20073

[58] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *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*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.). http://papers.nips.cc/paper_files/paper/2022/hash/

9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html

[59] Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. WebDancer: Towards Autonomous Information Seeking Agency. *CoRR* abs/2505.22648 (2025). arXiv:2505.22648 doi:10.48550/ARXIV.2505.22648

[60] Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Honglin Guo, Jiaqi Liu, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, Wei He, Yiwen Ding, Guanyu Li, Zehui Chen, Zhengyin Du, Xuesong Yao, Yufei Xu, Jiecao Chen, Tao Gui, Zuxuan Wu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. 2025. AgentGym-RL: Training LLM Agents for Long-Horizon Decision Making through Multi-Turn Reinforcement Learning. arXiv:2509.08755 [cs.LG] https://arxiv.org/abs/2509.08755

[61] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-Pack: Packed Resources For General Chinese Embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, Grace Hui Yang, Hongning Wang, Sam Han, Claudia Hauff, Guido Zuccon, and Yi Zhang (Eds.). ACM, 641–649. doi:10.1145/3626772.3657878

[62] Yang Xiao, Mohan Jiang, Jie Sun, Keyu Li, Jifan Lin, Yumin Zhuang, Ji Zeng, Shijie Xia, Qishuo Hua, Xuefeng Li, Xiaojie Cai, Tongyu Wang, Yue Zhang, Liming Liu, Xia Wu, Jinlong Hou, Yuan Cheng, Wenjie Li, Xiang Wang, Dequan Wang, and Pengfei Liu. 2025. LIMI: Less is More for Agency. arXiv:2509.17567 [cs.AI] https://arxiv.org/abs/2509.17567

[63] Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. 2025. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479* (2025).

[64] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jian Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 Technical Report. *CoRR* abs/2505.09388 (2025). arXiv:2505.09388 doi:10.48550/ARXIV.2505.09388

[65] Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards Thinking-Optimal Scaling of Test-Time Compute for LLM Reasoning. *CoRR* abs/2502.18080 (2025). arXiv:2502.18080 doi:10.48550/ARXIV.2502.18080

[66] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents. In *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*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.). http://papers.nips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html

[67] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629* (2022).

[68] Junjie Ye, Zhengyin Du, Xuesong Yao, Weijian Lin, Yufei Xu, Zehui Chen, Zaiyuan Wang, Sining Zhu, Zhiheng Xi, Siyu Yuan, Tao Gui, Qi Zhang, Xuanjing Huang, and Jiecao Chen. 2025. ToolHop: A Query-Driven Benchmark for Evaluating Large Language Models in Multi-Hop Tool Use. In *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*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, 2995–3021. https://aclanthology.org/2025.acl-long.150/

[69] Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. LIMO: Less is More for Reasoning. *CoRR* abs/2502.03387 (2025). arXiv:2502.03387 doi:10.48550/ARXIV.2502.03387

[70] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476* (2025).

[71] Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2024. AgentTuning: Enabling Generalized Agent Abilities for LLMs. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, 3053–3077. doi:10.18653/V1/2024.FINDINGS-ACL.181

[72] Yuyao Zhang, Zhicheng Dou, Xiaoxi Li, Jiajie Jin, Yongkang Wu, Zhonghua Li, Ye Qi, and Ji-Rong Wen. 2025. Neuro-Symbolic Query Compiler. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, 12138–12155. https://aclanthology.org/2025.findings-acl.628/

[73] Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2025. Group Sequence Policy Optimization. *CoRR* abs/2507.18071 (2025). arXiv:2507.18071 doi:10.48550/ARXIV.2507.18071

[74] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. DeepResearcher: Scaling Deep Research via Reinforcement Learning in Real-world Environments. *arXiv preprint arXiv:2504.03160* (2025).

Appendix

A Datasets

A.1 Training Data

We collected a diverse training dataset spanning four task categories to instill comprehensive agent capabilities.

- **General Tool-Use**: We sample 1k instances for labeled-tool scenarios and 1k for tool-retrieval from the ToolBench [37] training set. This data is intended to instill a generalized ability to use diverse tools and leverage large toolsets through retrieval.
- **Real-World Interaction**: We utilize 500 instances from ALFWorld [46] and 500 from WebShop [66], sampled from their training sets, to teach the model to interact effectively with environments, manage state transitions, and achieve user goals.
- **Deep Research**: We include 200 instances from WebDancer [59] and 500 from WebShaperQA [50] to enhance the model's proficiency in using web search and page browsing for in-depth information gathering.
- **Mathematical Reasoning**: We collect 0.9k problems from the DeepMath dataset [12] to strengthen the model's ability to use code as a tool for complex mathematical computations.

A.2 Benchmarks

We conduct extensive experiments on a wide range of benchmarks, including general tool-use and downstream applications.

General Tool-Use. These benchmarks encompass a broad range of distinct tools (from tens to over ten thousand), thus offering a testbed for evaluating different approaches to toolset scaling.

- **ToolBench [37]**: A large-scale benchmark containing over 16,000 real-world REST APIs spanning 49 categories. Test subsets include 100 test cases, designed to evaluate LLMs in both single-tool and complex multi-tool scenarios.
- **API-Bank [24]**: A comprehensive benchmark for tool-augmented LLMs. It features a runnable evaluation system with 73 API tools and a large training set (over 2,200 dialogues across 2,211 APIs from 1,008 domains), assessing LLMs' capabilities in planning, retrieving, and calling APIs.
- **TMDB [47]**: A sub-scenario of RestBench focused on the TMDB movie database, consisting of 100 questions that utilize 54 local tools and require an average of 2.3 sequential API calls.
- **Spotify [47]**: A sub-scenario of RestBench simulating a Spotify music player, featuring 57 questions and 40 local tools, demanding an average of 2.6 sequential API calls to complete the tasks.
- **ToolHop [68]**: A multi-hop reasoning dataset comprising 995 complex questions. It leverages 3,912 locally executable tools and requires between 3 to 7 sequential tool calls per task.

Downstream Applications. These benchmarks test the capability of different approaches in handling complex real-world tasks, which often require the use of domain-specific toolsets.

- **ALFWorld [46]**: A benchmark for simple Embodied AI tasks set in a text environment. Agents must complete objectives using a finite set of low-level embodied actions (eg., move, take) to test navigation and object manipulation.
- **WebShop [66]**: A challenging online shopping environment that provides 12,087 crowd-sourced tasks over a catalog of 1.18 million products. Agents interact with the simulated e-commerce website using core APIs: search[Query] and choose[Text Button].
- **GAIA [32]**: A complex benchmark for General AI Assistants, consisting of 466 real-world questions (with a 300-question held-out test set). It requires the flexible application of a broad general-purpose toolset including web browsing, code execution, multi-modal processing, and file handling.
- **Humanity's Last Exam (HLE) [35]**: A benchmark featuring 2,500 highly difficult, multi-disciplinary questions (graduate-level). It primarily evaluates the model's intrinsic deep reasoning and multi-modal understanding capabilities, as the questions are designed to be insoluble by simple external search tools.

B Baselines

We compare our proposed method with several baseline agents. The details of these baselines are introduced as follows:

- **ReAct (Reasoning and Acting) [67]**: ReAct is a general paradigm that combines reasoning and acting with language models. It prompts the model to generate a sequence of interleaved thought, action, and observation steps to solve a given task.
- **CodeAct [56]**: This is a framework where the agent's actions are expressed as Python code, which are then executed in an interpreter. By using code as the action space, the agent can interact with a wide variety of tools, APIs, and system functionalities.
- **Plan-and-Solve [54]**: This method follows a two-stage process to tackle complex problems. First, the model devises a detailed, step-by-step plan to solve the problem without using any tools. Then, it executes the plan, carrying out the necessary calculations or actions as outlined.
- **Reflexion [44]**: Reflexion is an approach that enhances agent learning through verbal self-reflection. After a failed attempt, the agent reflects on what went wrong and records this reflection in its memory.
- **AgentLM [71]**: An instruction tuning method designed to enhance the general agent capabilities of LLMs. It uses a lightweight, specially curated dataset called AgentInstruct to fine-tune LLMs.
- **WebThinker [26]**: WebThinker is a deep research agent designed for complex information-seeking tasks. It autonomously explores the web by interleaving a "think-search-write" process to gather and synthesize information.
- **HiRA (Hierarchical Reasoning Agent) [20]**: HiRA employs a hierarchical agent architecture to solve complex, multi-modal tasks. It decouples high-level planning from low-level execution by using a planner to decompose the task and executor agents to perform specific actions like searching or coding.
- **OpenAI Deep Research [34]**: This is a feature within ChatGPT that performs in-depth research on complex topics by exploring a wide range of web sources. It takes more time than a standard query to synthesize information and generate a detailed, structured response.

C Implementation Details

For DeepAgent, we use QwQ-32B [51] as the main reasoning model for the results in Table 1 and Table 2, and Qwen3-30B-A3B-Thinking-2507 [64] with Qwen3-235B-A22B-Thinking-2507 [64] in Table 5. We use Qwen2.5-32B-Instruct [40] as the auxiliary model for (1) filtering lengthy tool search results and execution outputs (this

Table 6: A case study of DeepAgent for multi-step, multi-tool tasks on ToolBench, showing only key actions.

Example of DeepAgent on a Multi-Tool Task
User Query: I'm organizing a film festival and I need assistance in finding the best films. Can you search for videos related to 'documentary' on Vimeo? Additionally, fetch the related people in the 'cinema' category to invite them as guest speakers. Finally, provide me with a streaming link for a YouTube video with the ID 'UxxajLWwzqY'.
DeepAgent Processing Output
Okay, I need to solve three tasks: find documentaries on Vimeo, find cinema-related people on Vimeo, and get a YouTube streaming link. I'll start by searching for the right tools. <tool_search> Vimeo video search for documentaries </tool_search> The search returned several tools. search_videos and get_videos_by_tag seem most relevant. I'll use the search tool first to find documentaries. <tool_call> {"name": "search_videos", "arguments": {"query": "documentary", "sort": "most_liked"}} </tool_call> The tool returned a list of popular documentary videos. Now for the second task: finding people in the 'cinema' category. Since there is no direct tool to search for people, I will try searching for videos with the 'cinema' tag and identify their creators. <tool_call> {"name": "get_videos_by_tag", "arguments": {"tag": "cinema", "sort": "most_liked"}} </tool_call> This call returned videos tagged with 'cinema' and included user information for each video, such as "John Doe" and "Jane Smith", who are potential guest speakers. Finally, for the YouTube streaming link, I'll first verify the video ID exists using the YouTube search tool. <tool_call> {"name": "search_for_youtube_v3", "arguments": {"part": "snippet", "type": "video", "q": "UxxajLWwzqY"}} </tool_call> The tool confirmed the video ID is valid. I can now construct the streaming link. I will now consolidate all the information to answer the user's query. **Final Answer:** 1. Documentaries on Vimeo: Found nature and space documentaries. 2. Cinema Guest Speakers: John Doe, Jane Smith with Vimeo profiles. 3. YouTube Link: https://www.youtube.com/watch?v=UxxajLWwzqY

is also applied to all baselines), (2) simulating RapidAPIs during ToolPO training, and (3) generating folded memory from interaction history. For the baselines, we use either QwQ-32B or Qwen2.5-32B-Instruct as the backbone model. Text generation for all models uses a maximum of 81,920 tokens, with a temperature of 0.7, top_p of 0.8, top_k of 20, and a repetition penalty of 1.05. The maximum number of actions is set to 50.

Web search and page browsing are implemented using the Google Serper API and Jina Reader API, respectively. The VQA tool is based on Qwen2.5-VL-32B-Instruct [1], which takes a question and an image as input and outputs a model-generated response. Tool retrieval is performed using bge-large-en-v1.5 [61]. All tool documentation follows the standard OpenAI function definition format: {"name": "...", "description": "...", "parameters": {"type": "object", "properties": {"param1": {"type": "...", "description": "..."}, ..., "required": ["param1"]}}. This format is used for building the toolset index and for all prompts given to the agents.

Training consists of 100 steps of ToolPO with a batch size of 64, $\lambda_1 = \lambda_2 = 1$, rollout size $K = 8$, and a maximum sequence length of 32,768. The maximum number of actions is 50. The training framework is based on VeRL [42] for multi-node distributed training. All experiments are conducted on 64 NVIDIA H20-141GB GPUs.

D Memory Schema

Our brain-inspired memory architecture consists of three components: episodic, working, and tool memory. To ensure stable memory folding and prevent information loss, each component is defined by a specific JSON schema. This structured format enables the agent to reliably parse and utilize the compressed memory, facilitating robust long-term reasoning.

Episodic Memory Schema. Episodic memory provides a high-level summary of the agent's task progression, major milestones, decisions, and outcomes. This allows the agent to maintain long-term context and reflect on its overall strategy. The format is: {"task_description": "A general summary of what the reasoning history has been doing and the overall goals it has been striving for.", "key_events": [{"step": "step number", "description": "A detailed description of the specific action taken, decision made, or milestone achieved at this step, including relevant context and reasoning behind the choice.", "outcome": "A detailed account of the direct result, observation, or feedback received from this action or decision, including any new information gained or changes in the task state."}], "current_progress": "A general summary of the current progress of the task, including what has been completed and what is left to be done."}

Working Memory Schema. Working memory functions as the agent's short-term buffer, holding information relevant to its immediate context. It focuses on the current sub-goal, active challenges, and planned next steps, ensuring continuity of reasoning across memory folds. The format is: {"immediate_goal": "A clear summary of the current subgoal—what you are actively working toward at this moment.", "current_challenges": "A concise summary of the main obstacles or difficulties you are presently encountering.", "next_actions": [{type": "tool_call or planning ordecision", description": "Anticipate and describe the next concrete action you intend to take to advance the task."}]}

Tool Memory Schema. Tool memory consolidates the agent's experiences with various tools. It tracks usage patterns, success rates, effective parameter combinations, and common errors. This structured knowledge enables the agent to learn from its interactions and refine its tool-use strategies over time. The format is: {"tools_used": [{"tool_name": "string", "success_rate": "float", "effective_parameters": ["param1", "param2"], "common_errors": ["error_type1", "error_type2"], "response_pattern": "description of typical output", "experience": "Reflect and summarize your experience, including both successes and failures."}], "derived_rules": ["When X condition occurs, prefer tool Y", "Tool Z works best with parameter A set to B"]}

E Case Study

To illustrate the effectiveness of our DeepAgent framework in handling complex, multi-step tasks that require coordinated use of multiple tools, we present a detailed case in Table 6. This example demonstrates how DeepAgent autonomously navigates tool selection, executes sequential actions, and synthesizes results to provide comprehensive solutions to user queries.