# AgentGym2: Benchmarking Large Language Model Agents in De-Idealized Real-World Environments

**Zhiheng Xi**[1,*,†], **Dingwen Yang**[1,*], **Jiaqi Liu**[1], **Jixuan Huang**[1], **Honglin Guo**[1], **Baodai Huang**[1], **Tinggang Chen**[1], **Qi Zhang**[1], **Zhonghang Lu**[1], **Chenyu Liu**[1], **Jiajun Sun**[1], **Jiazheng Zhang**[1], **Dingwei Zhu**[1], **Xin Guo**[1], **Junzhe Wang**[1], **Zhihao Zhang**[1], **Yuming Yang**[1], **Junjie Ye**[1], **Minghe Gao**[2], **Dongrui Liu**[3], **Jiaming Ji**[4], **Guohao Li**[5], **Tao Gui**[1,6,7,8,†], **Qi Zhang**[1,6,7], **Xuanjing Huang**[1,6,7,8]

[1]College of Computer Science and Artificial Intelligence, Fudan University,
[2]Zhejiang University, [3]Shanghai Jiao Tong University, [4]Peking University,
[5]CAMEL-AI.org, [6]Institute of Trustworthy Embodied AI, Fudan University,
[7]Shanghai Collaborative Innovation Center of Intelligent Visual Computing,
[8]Shanghai Innovation Institute

{zhxi22, tgui}@fudan.edu.cn

## Abstract

Language agents, i.e., LLM agents, progress rapidly and are increasingly deployed in production environments. This trend underscores the urgent need for rigorous and realistic evaluations. However, most existing benchmarks evaluate agents in simplified, idealized settings. They typically rely on pre-packaged tool interfaces, overlook critical steps, and assume inputs are clean and fully specified. Consequently, they understate the difficulty of real deployments, where uncertainty and noise are ubiquitous and agents must proactively explore the environment to uncover new tools. To bridge this gap, we present AgentGym2, a new evaluation framework with task instances grounded in real-world end-to-end working demands. Beyond reasoning and planning, it measures agents' ability to execute end-to-end procedures, discover tools via exploration, compose tools for unseen tasks, and remain robust to noisy and underspecified information. Experiments on 15 proprietary and open-source models show that even SOTA systems like Gemini and GPT-5 struggle on AgentGym2, revealing a substantial gap between the capability of current agents and the demands of real-world applications.

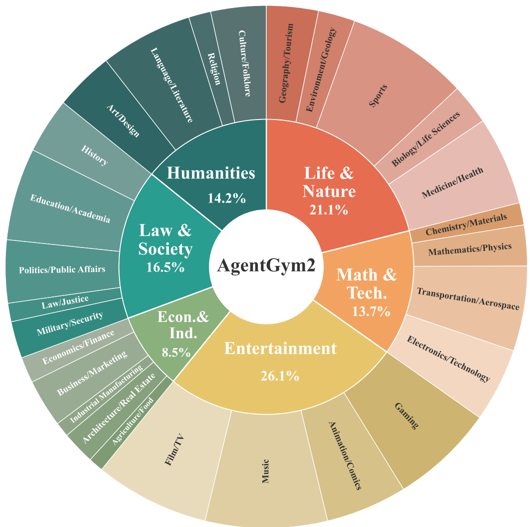


Figure 1: Domain distribution of AgentGym2. Econ. & Ind. denotes Economy & Industry. Math & Tech. denotes Math & Technology.

## 1 Introduction

As large language models advance rapidly, their applications evolve from simple conversational chatbots to autonomous agents capable of complex tasks such as deep research and data analysis, with increasing deployment in production (Li et al., 2025a; Zheng et al., 2025; Jin et al., 2025; Hong et al., 2025; Zhu et al., 2025). Consequently, it is essential to evaluate their ability in handling practical tasks in real-world (Garg et al., 2025; Patwardhan et al., 2025; Ko et al., 2026).

Despite rapid growth in agent evaluation research, existing benchmarks remains simplified or idealized (Xu et al., 2024; Patil et al., 2025), failing to capture the complexity in real-world environment. They typically provide pre-packaged tool sets, pre-solve intermediate yet necessary steps, and overlook the uncertainty and noise inherent in real-world scenarios. As a result, they may significantly underestimate real-world difficulty: user requests are often underspecified or ambiguous, information sources may be incomplete or misleading, and necessary tools may be unavailable upfront (Hathidara et al., 2025; Suri et al., 2025). Consequently, several capabilities essential for reliable production

*Equal contribution. †Corresponding authors.

[1]Our code and dataset are available at https://github.com/hotdog-zz/Agentgym2 and https://huggingface.co/datasets/hotdogzz/Agentgym2



*Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 44451–44479

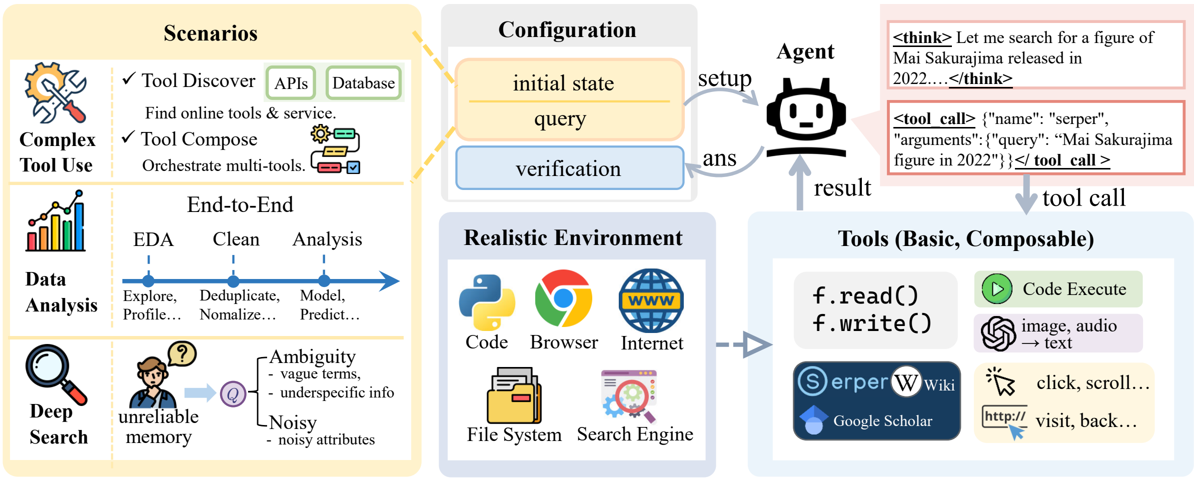


Figure 2: Overview of the AgentGym2 framework. It features a layered and modular design that decouples agents from tools and environments, comprising the following components: Environment, Tools, Configurations, and Agent. It includes three scenarios, and the tasks are grounded in non-idealized, complex real-world conditions.

| Scenario | Example Task | Explanation |
|---|---|---|
| Complex Tool Use | Please **obtain all available annual GDP data** (in US dollars) for China. Then, build an ARIMA model to forecast the GDP for 2023 and calculate the forecast error. | Models are required to discover and use **online World Bank** tools, **combining them with code** tools to solve the problem. |
| Data Analysis | Following the requirements in README.md, **clean and analysis** the given dataset, then answer: How much variance do the first two principal components explain, respectively? | Models are required to perform **end-to-end** data analysis covering data cleaning, EDA, and analysis. |
| Deep Search | Looking for a Japanese anime with monsters called "familiars"... they had white crystalline wings and could **throw crystal fragments**... People became monsters after making a pact—**trading** half of their soul... | The query contains **noise due to memory bias** (highlighted in **red**). Models must recognize these noisy attributes and avoid being misled to retrieve the correct answer. |

Figure 3: Examples of three distinct scenarios in AgentGym2. For complex tool using scenario, we focuses on the capability of LLM on tool discovery and composition. For data analysis, AgentGym2 challenges agents' end-to-end data analysis capabilities, including EDA procedures, data processing, and in-depth analysis. To simulate the noise in the real-world user, we consider the ambiguity and bias in deep search scenario.

deployment remain insufficiently evaluated.

To bridge this gap, we introduce AgentGym2, an evaluation framework for assessing language agents in realistic environments. It spans diverse scenarios—including data analysis, deep research and complex tool use—across over 27 domains. It emphasizes end-to-end task completion under real-world uncertainty, assessing not only standard agentic skills such as search (Wei et al., 2025; Zhou et al., 2025), planning (Erdogan et al., 2025), reasoning (Phan et al., 2025), and coding (Yang et al., 2024; Gao et al., 2025), but also an agent's ability to proactively clarify and interpret goals, explore and discover tools through interaction, compose tools into novel workflows, carry out complete procedures, and remain robust in the presence of noise, ambiguity, and underspecified inputs.

At the system level, AgentGym2 equips agents with a basic yet composable toolbox rather than query-specific, pre-selected tools (Yao et al., 2022; Liu et al., 2024; Xi et al., 2025a). This design exposes a large and realistic action space, encouraging agents to explore what to use and how to use it. The toolbox covers five core categories—web browsing, information retrieval, file processing, multimodal understanding, and code execution—supporting over 27 actions. Built with a modular, decoupled architecture, AgentGym2 enables environment isolation and parallel tool invocation, ensuring rigorous, reproducible, and scalable evaluation.

To ground our task instances in authentic user demands, we curate real requests from diverse platforms, including GitHub, Reddit, and Kaggle. We transform these raw needs into challenging and diverse evaluation instances through a combination of automated synthesis, query augmentation, and manual annotation. To ensure quality and consistency at scale, all tasks undergo a cascading verification pipeline that combines model-based checks with human review, validating task validity, clarity, and expected outcomes.

We evaluate 15 leading proprietary and open-

| Benchmark | Scenario | Non-Simulated Environment? | Realistic Action Space? | Tool discovery? | E2E Data Analysis? | Inserted Distractors? |
|---|---|---|---|---|---|---|
| BrowseComp (Wei et al., 2025) | Search | yes | yes | no | - | no |
| DA-Code (Huang et al., 2024) | Data Analysis | yes | yes | no | no | no |
| $\tau$-Bench (Yao et al., 2025) | Tool Use | no | yes | no | - | no |
| AgentBoard (Ma et al., 2024) | Multi-Scenarios | no | no | no | - | no |
| AgentBench (Liu et al., 2024) | Multi-Scenarios | no | no | no | no | no |
| AgentGym (Xi et al., 2025a,b) | Multi-Scenarios | no | no | no | - | no |
| AgentGym2 (Ours) | Multi-Scenarios | yes | yes | yes | yes | yes |

Table 1: Comparison of different agent benchmarks. Specifically, Tool discovery indicates whether the necessary tools or resources are not provided at the start and must be discovered during interaction; E2E Data Analysis (incl. cleaning) indicates whether the benchmark covers an end-to-end data analysis pipeline, including data cleaning/preprocessing and analysis; and Distractors indicates whether the tasks include explicitly injected distracting information, such as noisy information in search queries or dirty/irrelevant content in data files. Our Agentgym2 is the only benchmark that roots in realistic environment and action space with all of the three dimensions.

source models on AgentGym2 and find that even state-of-the-art systems such as GPT-5 (~44%) and Claude Sonnet 4.5 (~37%) fail to deliver satisfactory performance, indicating that today's agents still require substantial progress before they can be reliably deployed in production. For open-source models, strong post-training can significantly boost performance: Nex-N1-32B and Nex-N1-671B (Nex-AGI et al., 2025) outperform their respective base models by roughly 9.16% and 10.53%. We also observe consistent gains as model parameters and computational cost scale, whereas increasing the number of interaction turns does not produce a scaling effect. Fine-grained analyses in Section 6 characterize agent behaviors and reveal recurring failure patterns across scenarios, such as insufficient exploration (Xi et al., 2025b) and hallucinations (Sahoo et al., 2024).

In summary, our main contributions are:

- We propose a new interaction framework that equips agents with a basic action space and supports further exploration, enabling isolated, parallel, and scalable interactions.
- We introduce a new benchmark to evaluate LLM agents under de-idealized environments and realistic requirements.
- We conduct extensive experiments and analyses to provide in-depth insights, aiming to support future research, development, and deployment of language agents.

## 2 Related Works

**Development of language agents.** Early large language models are primarily framed as conversational systems designed for dialogue. This paradigm shifts with the introduction of approaches such as ReAct (Yao et al., 2023), which enables language agents to iteratively interact with complex environments through action and feedback. Building on this foundation, a growing body of work develops domain-specialized agents spanning software engineering (Yang et al., 2024; Wang et al., 2025a), web navigation and information seeking (Jin et al., 2025; Zheng et al., 2025; Li et al., 2025a), and data analysis pipelines (Hong et al., 2025; Guo et al., 2025a; Zhu et al., 2025), highlighting the versatility of agent-based frameworks and their promise for practical deployment across a range of real-world tasks. Despite strong results on curated benchmarks, existing agent architectures rely on idealized, closed-world assumptions with pre-specified tools, limiting their effectiveness in unconstrained, real-world workflows.

**Benchmarks for evaluating language agents.** Recent benchmarks have begun to evaluate language agents in richly simulated environments that span a wide range of interactive tasks, including web navigation and shopping (Yao et al., 2022), embodied and household tasks (Shridhar et al., 2021), scientific discovery environments (Wang et al., 2022), text-based games (Prasad et al., 2024), and tool-using scenarios (Patil et al., 2025; Yao et al., 2025; Barres et al., 2025). By integrating these heterogeneous tasks under a unified interaction interface and evaluation protocol, frameworks such as AgentBench (Liu et al., 2024) and AgentGym (Xi et al., 2025a) provide a standardized testbed for assessing agent behavior across diverse environments. Beyond such wide-spectrum platforms, the community has also developed a growing number of domain-specific benchmarks tailored to real-world application areas, most notably software engineering (Yang et al., 2025b; Deng et al., 2025; Zan et al., 2025; Terminal-Bench, 2025), data science workflows (Lai et al., 2023; Lei et al., 2025; Wang et al., 2025c; Egg et al., 2025) and composing multiple well-formatted real MCP tools (Wang et al., 2025b; Mo et al., 2025; Guo et al., 2025b). Despite these advances, existing environments often rely on sandboxed settings with curated tools and well-specified instructions, which limits their ability to assess open-ended tool discovery and robustness. We address this limitation with a non-idealized evaluation suite focused on end-to-end execution and dynamic tool acquisition. We provide a comparison on core dimensions in real deployment settings in Table 1.

# 3 AgentGym2 Framework

## 3.1 Architecture Overview

AgentGym2 adopts a layered and modular design that decouples agents from tools and environments (Li et al., 2023; Xi et al., 2025a; Ko et al., 2026). As illustrated in Figure 2, the architecture comprises the following components:

- **Environment**: The realistic digital runtime that provides underlying infrastructure and services for agents to execute tasks.
- **Tools**: The interfaces exposed by environments. In AgentGym2, these are general-purpose, basic, and composable tools.
- **Agent**: The LLM agent that receives observations from environments via the interfaces, performs reasoning over its interaction trajectory, and outputs actions. Note that an agent may invoke multiple tools simultaneously within a single step.
- **Configurations**: Composed of a user request, a specified initial state, and the corresponding task verification mechanism. Inputs for agents encompass multiple modalities, including text and images, and a single scenario may involve multiple environments.

## 3.2 Features and Characteristics

AgentGym2 extends AgentGym (Xi et al., 2025a,b), which provides **simulated** environments and corresponding **specialized** interfaces for LLM agents. In this work, we extend it with realistic digital environments (Ko et al., 2026), construct a basic, general-purpose, and composable toolbox, and implement sophisticated engineering designs to support scalable evaluation in real-world settings.

**Realistic scenarios and environments.** We first identify three high-demand real-world scenarios for agentic tasks: complex tool usage, data analysis, and deep search. These demand high-level reasoning and the ability to handle heterogeneous data. AgentGym2 supports these requirements by providing a reliable infrastructure across five domains: code execution, file systems, search engines, web browsers and internet. We then synthesize user requests grounded in these real-world task distributions, requiring agents to interact with multimodal inputs like webpage snapshots and source code. This approach ensures our benchmark reflects practical challenges rather than environment-specific limitations (Liu et al., 2024; Xi et al., 2025a).

**Basic and composable toolbox.** Instead of preselecting domain-specific tools, AgentGym2 provides a basic, general-purpose, and composable toolbox covering five core categories: web browsing, information retrieval, file processing, multimodal understanding, and code execution. We achieve this by collecting a suite of real-world tools, fixing their existing bugs, and extending them to support isolated runtime, ultimately integrating them into a unified toolbox. Notably, we grant agents full access to the entire internet rather than limiting them to pre-restricted websites, enabling them to actively discover and learn the usage of additional tools as needed. This design provides agents with a large and realistic action space.

**Reliable engineering design.** To ensure scalable, precise, and reproducible evaluation, AgentGym2 implements sophisticated engineering optimizations. First, it ensures strict runtime isolation to prevent erroneous evaluations, including file management systems to avoid read-write conflicts, isolated browser instances to prevent history-based cheating, and sandboxed environments for safe code execution (Yang et al., 2025b; Pan et al., 2025). Moreover, AgentGym2 supports parallel tool invocation via asynchronous execution, enabling agents to initiate multiple tool calls concurrently within a single interaction. This design allows simultaneous querying of multiple sources or parallel execution of tasks, ensuring efficient resource use.

# 4 Benchmark Construction

Besides basic agentic skills such as reasoning, planning, and coding, AgentGym2 targets a broader set of capabilities: **proactively clarifying and**

| Statistics | Number |
|---|---|
| Total query | 437 |
| - Complex Tool Use | 182 |
| - Data Analysis | 57 |
| - Deep Search | 198 |
| Language | ZH/EN |
| Total Attachment | 652 |
| Attachment Modality | Text, Image, Audio, Video |
| Basic Tool Types | 27 |

Table 2: Key statistics of AgentGym2.

**interpreting goals, discovering tools through interaction, composing tools into novel workflows, executing complete procedures end to end, and remaining robust to noise, ambiguity, and underspecified inputs**.

We next describe our data construction methodology. In total, AgentGym2 covers 437 tasks spanning 27 domains, the distribution is in Figure 1. The key statistics is in Table 2, and the cases of three scenarios can be seen in Figure 3.

**Complex tool using scenario.** We first curate high-frequency web tools across academic, daily-life, and entertainment domains. Based on these tools, we focus on constructing two types of instances: tool discovery and tool composition, which contain 57 and 125 instances, respectively. Our instances are created through (i) manually adapting real-world discussions and questions from volunteers from diverse backgrounds, and (ii) augmenting existing benchmarks (e.g., BrowseComp (Wei et al., 2025)) by having our authors replace key query entities with external attachments, such as CSV/JSON files or manually curated images, which may include distractors. A rigorous, iterative verification process ensures that each task is solvable using the provided basic tools—potentially by discovering additional interfaces or composing novel workflows—and that reaching the solution requires processing the attachments and yields a unique, verifiable answer.

**Data analysis scenario.** Previously, data analysis benchmarks typically provide preprocessed, high-quality data and focused only on final outcomes, while overlooking necessary steps such as Exploratory Data Analysis (EDA) and data cleaning (Lai et al., 2023; Wang et al., 2025c; Egg et al., 2025). As a result, the evaluation conditions are inconsistent with real-world production settings. Hence, our authors construct 57 instances based on sources such as GitHub and Kaggle by enriching real-world problems, injecting target challenges, and manually building tasks from scratch. These tasks challenge agents' end-to-end data analysis capabilities, including EDA procedures, data processing, and in-depth analysis. A quality control step similar to the one used in complex tool use was also conducted to ensure data quality.

**Deep search scenario.** Deep Search requires iterative retrieval, synthesis, denoising, and analysis across multiple sources (Wei et al., 2025; Zhou et al., 2025; Jin et al., 2025). Real queries can be well-specified, but often include ambiguity or memory-induced inaccuracies—cases underrepresented in prior benchmarks. We therefore generate two query types via controlled perturbations: (i) Ambiguity-only (114 instances) with entity/relational ambiguity; and (ii) Ambiguity-with-bias (84 instances) combining ambiguity with inaccurate attributes. This setting evaluates both core deep-search skills (retrieval, multi-hop reasoning, cross-source synthesis) and robustness to underspecified, irrelevant, and noisy information. Ambiguity-only instances are automatically synthesized with quality control by selecting a Wikipedia answer entity, recursively sampling relations to introduce ambiguity, merging them into multi-hop questions, and verifying difficulty and answer uniqueness. Ambiguity-with-bias instances are sourced from memory-based social-media queries (e.g., Reddit, Xiaohongshu) or are augmented by injecting noisy attributes with hedging phrases to ambiguity-only instances; all instances are manually checked to ensure the unique answer remains unchanged.

**Answer verify.** All reference answers in AgentGym2 consist of short strings or numbers. This format enables straightforward evaluation through LLM-based judges that assess semantic equivalence between predicted and reference answers.

## 5 Experiments

### 5.1 Experimental Setups

**Models.** We evaluate both proprietary and open-source models. Proprietary models include GPT-5, Claude-4.5-Sonnet, and Gemini-2.5-Pro (Gemini-Team, 2025). Open-source models include Kimi-K2 (Bai et al., 2025), Qwen3 series (Qwen3-8B, Qwen3-32B, Qwen3-235B-A22B-2507) (Yang et al., 2025a), Nex-N1 series (Nex-N1-32B, Nex-N1-671B) (Nex-AGI et al., 2025), GLM-4.6 (Zeng

| Models | Thinking | Complex Tool Usage | | Data Analysis | | Deep Search | | Overall | |
|---|---|---|---|---|---|---|---|---|---|
| | | Avg@3 | Pass@3 | Avg@3 | Pass@3 | Avg@3 | Pass@3 | Avg@3 | Pass@3 |
| *Open-sourced Models < 100B* | | | | | | | | | |
| Qwen3-8B | ✗ | 2.75 | 7.14 | 0.00 | 0.00 | 6.57 | 12.63 | 4.12 | 8.70 |
| Qwen3-8B | ✓ | 4.58 | 9.34 | 1.75 | 5.26 | 7.41 | 13.13 | 5.49 | 10.53 |
| Qwen3-32B | ✗ | 6.21 | 13.19 | 1.75 | 5.26 | 8.25 | 16.16 | 6.55 | 13.50 |
| Qwen3-32B | ✓ | 6.41 | 12.64 | 2.34 | 7.02 | 9.76 | 19.70 | 7.40 | 15.11 |
| Nex-N1-32B | ✗ | **14.65** | **26.92** | **9.36** | **19.30** | **18.52** | **35.35** | **15.71** | **29.75** |
| *Open-sourced Models ≥ 100B* | | | | | | | | | |
| Qwen3-235B-A22B-Instruct | ✗ | 12.27 | 21.98 | 9.94 | 19.30 | 16.84 | 30.30 | 14.04 | 25.40 |
| Qwen3-235B-A22B-Think | ✓ | 7.33 | 12.09 | 2.92 | 7.02 | 13.97 | 22.73 | 9.76 | 16.25 |
| Kimi-K2 | ✗ | 19.96 | 34.62 | 16.96 | 31.58 | 22.90 | 39.39 | 20.90 | 36.38 |
| DeepSeek-V3.1 | ✗ | 23.26 | **43.41** | 20.47 | 36.84 | 20.54 | 38.38 | 21.66 | 40.27 |
| Deepssek-V3.2-Exp | ✗ | 19.60 | 39.56 | 13.45 | 31.58 | 24.41 | 45.96 | 20.98 | 41.42 |
| GLM-4.6 | ✓ | 22.16 | 39.01 | 8.19 | 19.30 | 22.56 | 39.40 | 20.52 | 36.62 |
| Nex-N1-671B | ✗ | **29.30** | 42.86 | **29.82** | **49.12** | **35.53** | **53.55** | **32.19** | **48.52** |
| *Proprietary Models* | | | | | | | | | |
| Gemini-2.5-Pro | ✓ | 21.25 | 36.81 | 12.28 | 24.56 | 22.56 | 36.87 | 20.67 | 35.24 |
| Claude-4.5-Sonnet | ✓ | 42.38 | 52.20 | **39.77** | 50.88 | 31.63 | 42.91 | 37.17 | 47.82 |
| GPT-5 | ✓ | **48.72** | **67.58** | 39.18 | **59.65** | **45.80** | **65.66** | **46.15** | **65.68** |

Table 3: Performance comparison against different LLMs on AgentGym2. For each group, the best result is in **bold**, and the second-best is underlined.

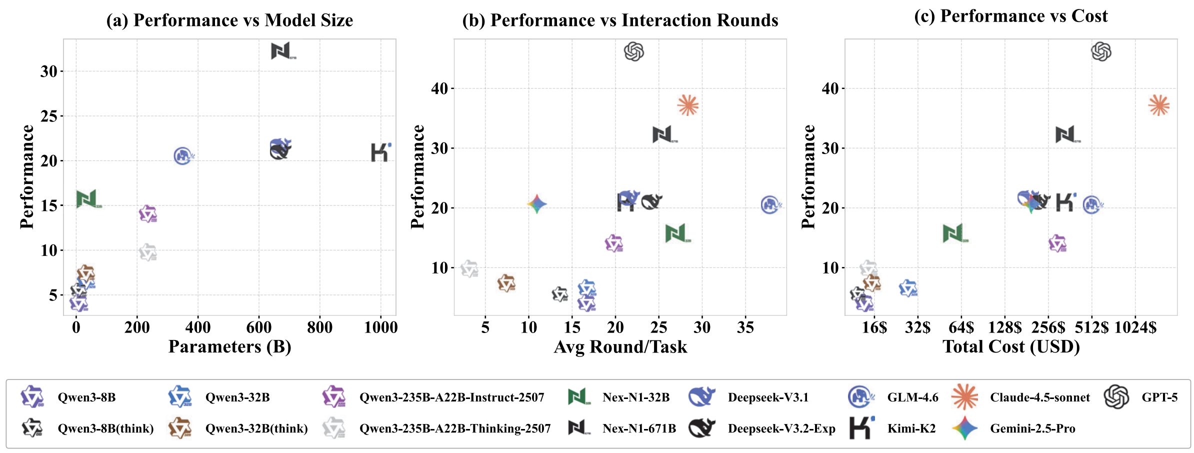


Figure 4: Effect on performance across three dimensions: total cost, the average number of interaction rounds, and the parameter scale.

et al., 2025), and DeepSeek series (DeepSeek-V3.1, DeepSeek-V3.2-Exp) (DeepSeek-AI, 2024; DeepSeek-AI et al., 2025).

**Implementation details.** We leverage the ReAct (Yao et al., 2023) style for agents, with all tools provided in the OpenAI tool schemas. We limit the number of interaction rounds to 100, and the task terminates when the agent outputs an answer enclosed within the special format "<final_answer>...</final_answer>". For the main results, each task is run three times with a fixed temperature of 0.6 and a maximum generation length of 8192 tokens of each round. The LLM judge model is implemented with Qwen3-235B-A22B-Instruct-2507, more details are provided in Appendix D.

## 5.2 Main Result

**AgentGym2 poses significant challenges for current models, especially for open-source series.** The evaluation results are illustrated in Table 3. Both proprietary and open-source models struggle with AgentGym2. Even GPT-5 achieves only 46.15 on Avg@3, while the top-performing open-source agents Nex-N1-671B and DeepSeek-V3.1 achieve merely 32.19 and 21.66, respectively. This demonstrates that AgentGym2 presents a significant challenge for current SOTA models, revealing a substantial gap between the capabilities of existing agents and the demands of real-world deployments.

**Model performance varies across different scenarios.** Most models underperform in Data Analysis scenario compared to other scenarios. For instance, GLM-4.6 achieves 22.16 on Complex Tool Use but only 8.19 on Data Analysis, a drop of 13.97 points. Claude-4.5-Sonnet and the Nex-N1 series maintain much more stable performance in Data Analysis, which can be attributed to their strong agentic coding capabilities, empowering them to parse complex dependencies between data entries and multimodal files, thereby leading to higher task success rates.

**Agentic post-training improves model performance.** Nex-N1-32B (post-trained from Qwen3-32B) and Nex-N1-671B (post-trained from DeepSeek-V3.1) outperform their respective base models by approximately 9.16 and 10.5 in Avg@3. These gains underscore the efficacy of agentic post-training and suggest that it significantly narrows the performance gap with proprietary systems.

### 5.3 Scaling Trends with Model size, Interaction Rounds, and Budget

In Figure 4, we explore the effect of model performance across three key dimensions: parameter scale, total cost, and the average number of interaction rounds.

**Performance exhibits a clear upward trend correlated with model scale.** Within the Qwen3 series, the 8B, 32B, and 235B (non-thinking) variants achieve scores of 8.70, 10.53, and 25.40, respectively. This suggests that scaling model size remains an effective dimension for improving performance and is worth further investigation.

**High-performing models tend to exhibit a moderate number of interaction rounds per task.** Top-tier models, such as GPT-5, Claude-4.5-Sonnet, and Nex-N1-671B, typically operate within 20–30 interaction rounds per task. Notably, reasoning-oriented (thinking-mode) variants require fewer interactions than non-thinking modes, as they prioritize internalized deliberation over frequent external tool invocation. Specifically, Qwen3-235B-A22B-Thinking-2507 exhibits the lowest interaction frequency among all competitors; however, its performance lags behind the instruct version by 5.31 points. This performance gap indicates that balancing reasoning and interaction is crucial for optimizing performance in real-world tasks.

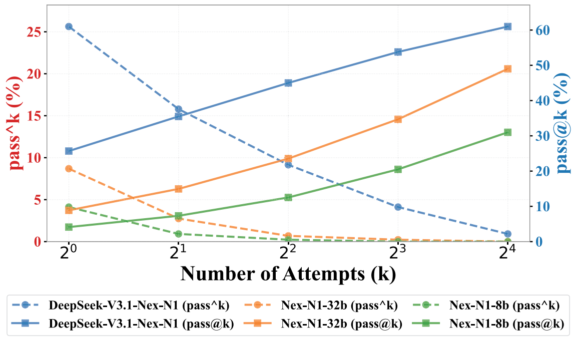


Figure 5: Pass@k and Pass^k performance on AgentGym2. Pass@k measures the probability that at least one correct solution appears among the sampled trajectories, whereas Pass^k measures the consistency that all of the k sampled trajectories are correct.

**Cost-performance Trade-offs.** Top-performing models typically come with high costs, and practitioners must balance performance, efficiency, and budget. This trade-off suggests that current agents still fall short of real-world production needs, where both strong performance and cost efficiency are essential.

## 6 Analysis and Discussion

**LLM-as-judge reliability.** We measure human–LLM agreement during our evaluation. Specifically, we manually inspected all outputs from four representative models—Gemini-2.5-Pro, GPT-5, Claude-4.5-Sonnet, and Nex-N1-671B—and compared each agent output against the reference answer. The overall agreement between the LLM judge and human verification is 98.9%, supporting the reliability of the LLM-as-judge.

**Pass@k and Pass^k performance.** We investigate the effect of test-time compute scaling on a subset of AgentGym2 (100 instances) using the Nex-N1 series models. Specifically, we sample multiple trajectories for each query and report their performance under sampling budget $N$. As shown in Figure 5, increasing test-time compute improves Pass@k, indicating that more tasks can be solved. For example, the high-capacity Nex-N1-671B gains up to ~30% with larger sampling budgets. In contrast, the Pass^k metric declines sharply because it requires success in all $N$ samples, highlighting the model's limited stability and robustness in deployment.

**The importance of tool discovery and downstream procedures.** In AgentGym2, tool discovery is a key evaluation focus, and we analyze this

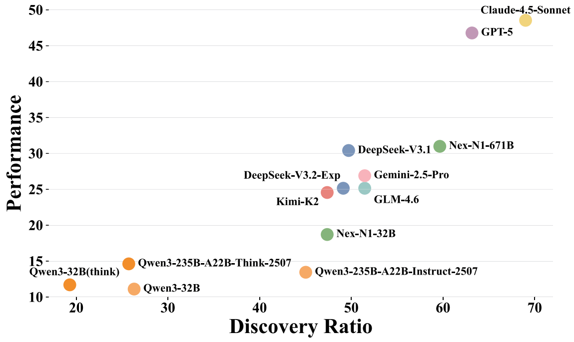


Figure 6: Relationship between discovery ratio and performance on complex tool using. The discovery ratio denotes the proportion of tasks in which agents successfully discover relevant online tools.

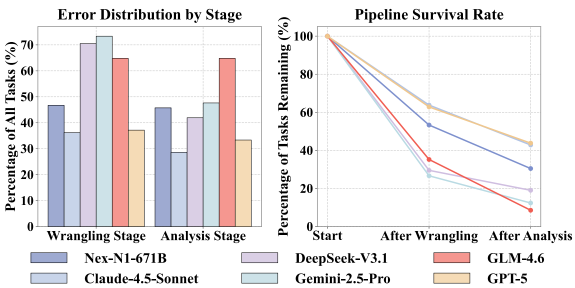


Figure 7: Error distribution (left) and survival rates (right) across the data wrangling and analysis stages.

subset in depth. We define the discovery ratio as the proportion of agents that successfully identify relevant specialized online tools. As shown in Figure 6, discovery ratio correlates strongly with task performance, indicating that tool discovery is a prerequisite for success. Yet performance remains limited even when the right tools are found, suggesting that downstream abilities—such as tool composition and correct invocation—are equally critical and underscoring the need for end-to-end competence.

**Data wrangling poses significant challenges.** In AgentGym2, we argue that effective data analysis requires both data wrangling and downstream analysis. Wrangling tests robustness through dataset exploration, validation, and automatic repair (e.g., missing values and schema inconsistencies), while the analysis stage requires careful reasoning and insight extraction to answer the task.

Hence, we examine the error distribution and survival rates across the data wrangling and analysis stages, as illustrated in Figure 7. The left panel reveals that data wrangling constitutes the primary bottleneck in data analysis scenarios. The right panel shows that the survival rate drops most sharply after data wrangling, suggesting that future agent development should account for complex real-world environments and full end-to-end workflows.

**Impact of noisy information on deep search performance.** Real users' deep search queries are often noisy and may even reflect memory biases (Wang et al., 2024; Java et al., 2025; Li et al., 2024), and agent performance in such settings is crucial for user experience. To simulate this, we inject noise into otherwise correct queries and evaluate models with and without the added noise. As shown in Figure 8, introducing noisy information causes substantial performance drops across all models, with Claude-4.5-Sonnet decreasing by 6.4% and GPT-5 by 7.4%. This indicates that current agents remain insufficiently robust to noisy information in real-world queries.

**Failure mode analysis.** We conduct a fine-grained error analysis across all scenarios, using GPT-5 to categorize failures according to the taxonomy in Table 7 in Appendix F.1, where we provide detailed descriptions of each error type.

As shown in Figure 9, overall, the model exhibits different failure patterns across tasks; however, incorrect analysis (24.0% on average) and insufficient exploration (22.8% on average) consistently emerge as the leading error modes across scenarios. This suggests that under real-world conditions, the model's reasoning/analysis capability and its ability to explore the open world remain key weaknesses. In complex tool use, failures are primarily characterized by confirmation bias (24.7%), where the model is overconfident and relies on internal priors rather than actively discovering or invoking external tools for verification. In data analysis, instruction misinterpretation is the dominant failure mode (27.0%), indicating difficulty in maintaining precise alignment with fine-grained task constraints. In deep search, errors are largely driven by insufficient exploration (35.2%), with the model prematurely terminating search trajectories in noisy and ambiguous environments. Collectively, these patterns reveal limited robustness in sustaining multi-turn decision processes and insufficient utilization of environmental feedback, which remain key bottlenecks for deploying LLM agents in complex, real-world tasks.

We further present detailed case studies of representative failure modes in Appendix F.2 (Figures 14–21).

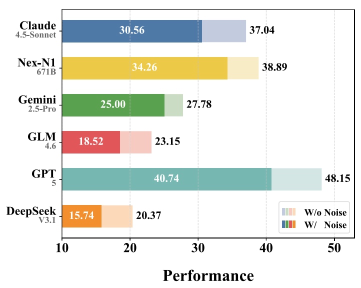


Figure 8: Effect of the added noise in query on deep search.

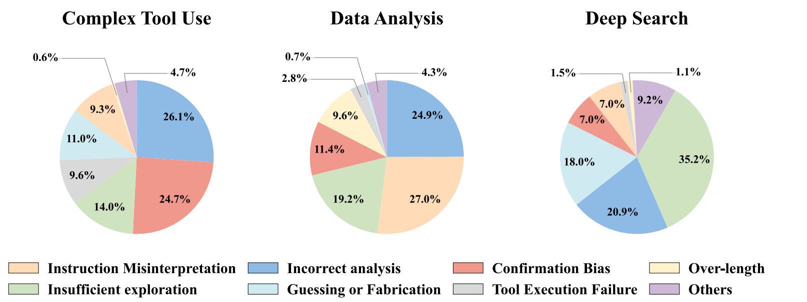


Figure 9: Failure model analysis across all scenarios, where incorrect analysis and insufficient exploration consists of the main error modes.

| Models | Hidden-information retrieval | | Tool Discovery | | Ambiguity/noise with bias | |
|---|---|---|---|---|---|---|
| | idealized | de-idealized | idealized | de-idealized | idealized | de-idealized |
| Gemini-2.5-Pro | 33.33 | 18.67 | 33.33 | 26.90 | 27.78 | 25.00 |
| GPT-5 | 61.67 | 49.60 | 53.57 | 46.78 | 48.15 | 40.74 |

Table 4: Model performance in de-idealized factor settings. All results are reported in Avg@3.

**Ablation study on de-idealized factors.** We select subsets and conduct controlled ablations on key de-idealized factors to quantify their impact on model performance.

- **Hidden-information retrieval**: In some tasks, we place important information or clues inside files, and the agent must read the files and filter out irrelevant content (distractors). Here, we ablate this factor by adopting an idealized setting in which we directly provide the agent with this information.

- **Tool discovery**: In some tasks, agents need to discover new tools to solve the task. Here, we ablate this factor by adopting an idealized setting where the required tools are provided to the agent directly.

- **Ambiguity/noise with bias**: In some tasks, we include noisy content, such as bias introduced by a user's memory. Here, we ablate this factor by adopting an idealized setting that removes such noise and biases.

The results in Table 4 show that performance improves substantially after ablation, indicating that these common real-world non-ideal factors are key contributors to performance degradation. By explicitly incorporating them, AgentGym2 better reflects the requirements of real deployment.

## 7 Conclusion

In this work, we introduced AgentGym2, a new framework to evaluate language agents in de-idealized real-world settings. Beyond reasoning and planning, AgentGym2 measures agents' ability to execute end-to-end procedures, discover tools via exploration, compose tools for unseen tasks, and remain robust to noisy and underspecified information. We implement the whole framework by equipping models with basic and composable tools, enabling isolated, parallel, and scalable interactions. Our extensive experiments across 27 diverse domains show that AgentGym2 remains highly challenging for current state-of-the-art models. We hope this work will help the community build more robust agents that can meaningfully improve real-world productivity.

## Limitations

While AgentGym2 represents a significant step towards de-idealized agent evaluation, our work has several limitations. First, although we cover diverse domains, our benchmarks rely primarily on public web interfaces and general-purpose tools, omitting proprietary enterprise software (e.g., internal CRMs or legacy systems) that is prevalent in actual production environments. Second, our current metrics focus exclusively on functional capabilities and task success; we do not explicitly evaluate safety and security aspects, such as data leakage risks and vulnerability to adversarial attacks, which are prerequisites for reliable deployment. Finally, due to the substantial computational resources required for multi-turn agent interactions, our experiments are constrained in scale and granularity, preventing us from performing more exhaustive ablations across a wider range of hyperparameters or prompt strategies.

## Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 62576106, 62521004, 62476061, 62376061).

## References

Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, Yichen Feng, and 80 others. 2025. Kimi K2: open agentic intelligence. *CoRR*, abs/2507.20534.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. 2025. $\tau^2$-bench: Evaluating conversational agents in a dual-control environment. *CoRR*, abs/2506.07982.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *CoRR*, abs/2412.19437.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 245 others. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *Preprint*, arXiv:2512.02556.

Xiang Deng, Jeff Da, Edwin Pan, Yannis Yiming He, Charles Ide, Kanak Garg, Niklas Lauffer, Andrew Park, Nitin Pasari, Chetan Rane, Karmini Sampath, Maya Krishnan, Srivatsa Kundurthy, Sean Hendryx, Zifan Wang, Chen Bo Calvin Zhang, Noah Jacobson, Bing Liu, and Brad Kenstler. 2025. Swe-bench pro: Can AI agents solve long-horizon software engineering tasks? *CoRR*, abs/2509.16941.

Alex Egg, Martin Iglesias Goyanes, Friso Kingma, Andreu Mora, Leandro von Werra, and Thomas Wolf. 2025. Dabstep: Data agent benchmark for multi-step reasoning. *CoRR*, abs/2506.23719.

Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-act: Improving planning of agents for long-horizon tasks. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

Pengfei Gao, Zhao Tian, Xiangxin Meng, Xinchen Wang, Ruida Hu, Yuanan Xiao, Yizhou Liu, Zhao Zhang, Junjie Chen, Cuiyun Gao, Yun Lin, Yingfei Xiong, Chao Peng, and Xia Liu. 2025. Trae agent: An llm-based agent for software engineering with test-time scaling. *CoRR*, abs/2507.23370.

Divyansh Garg, Shaun VanWeelden, Diego Caples, Andis Draguns, Nikil Ravi, Pranav Putta, Naman Garg, Tomas Abraham, Michael Lara, Federico Lopez, James Liu, Atharva Gundawar, Prannay Hebbar, Youngchul Joo, Jindong Gu, Charles London, Christian A. Schröder de Witt, and Sumeet Motwani. 2025. REAL: benchmarking autonomous agents on deterministic simulations of real websites. *CoRR*, abs/2504.11543.

Gemini-Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, abs/2507.06261.

Honglin Guo, Kai Lv, Qipeng Guo, Tianyi Liang, Zhiheng Xi, Demin Song, Qiuyinzhe Zhang, Yu Sun, Kai Chen, Xipeng Qiu, and Tao Gui. 2025a. CritiQ: Mining data quality criteria from human preferences. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16240–16261, Vienna, Austria. Association for Computational Linguistics.

Zikang Guo, Benfeng Xu, Chiwei Zhu, Wentao Hong, Xiaorui Wang, and Zhendong Mao. 2025b. Mcp-agentbench: Evaluating real-world language agent performance with mcp-mediated tools. *Preprint*, arXiv:2509.09734.

Ashutosh Hathidara, Julien Yu, and Sebastian Schreiber. 2025. Disambiguation-centric finetuning makes enterprise tool-calling llms more realistic and less risky. *CoRR*, abs/2507.03336.

Sirui Hong, Yizhang Lin, Bang Liu, Bangbang Liu, Binhao Wu, Ceyao Zhang, Danyang Li, Jiaqi Chen, Jiayi Zhang, Jinlin Wang, Li Zhang, Lingyao Zhang, Min Yang, Mingchen Zhuge, Taicheng Guo, Tuo Zhou, Wei Tao, Robert Tang, Xiangtao Lu, and 9 others. 2025. Data interpreter: An LLM agent for data science. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 19796–19821, Vienna, Austria. Association for Computational Linguistics.

Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, Zeyu Zhang, Yifeng Wang, Qianshuo Ye, Bernard Ghanem, Ping Luo, and Guohao Li. 2025. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation. *Preprint*, arXiv:2505.23885.

Yiming Huang, Jianwen Luo, Yan Yu, Yitong Zhang, Fangyu Lei, Yifan Wei, Shizhu He, Lifu Huang, Xiao

Liu, Jun Zhao, and Kang Liu. 2024. Da-code: Agent data science code generation benchmark for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 13487–13521. Association for Computational Linguistics.

Abhinav Java, Ashmit Khandelwal, Sukruta Prakash Midigeshi, Aaron Halfaker, Amit Deshpande, Navin Goyal, Ankur Gupta, Nagarajan Natarajan, and Amit Sharma. 2025. Characterizing deep research: A benchmark and formal definition. *CoRR*, abs/2508.04183.

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.

Dogyu Ko, Chanyoung Yeo, Daeho Kim, Jaeho Kim, and Hyoseok Hwang. 2026. GAIA: generating task instruction aware simulation grounded in real contexts using vision-language models. *IEEE Robotics Autom. Lett.*, 11(1):626–633.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-Tau Yih, Daniel Fried, Sida I. Wang, and Tao Yu. 2023. DS-1000: A natural and reliable benchmark for data science code generation. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 18319–18345. PMLR.

Fangyu Lei, Jinxiang Meng, Yiming Huang, Junjie Zhao, Yitong Zhang, Jianwen Luo, Xin Zou, Ruiyi Yang, Wenbo Shi, Yan Gao, Shizhu He, Zuo Wang, Qian Liu, Yang Wang, Ke Wang, Jun Zhao, and Kang Liu. 2025. Dacomp: Benchmarking data agents across the full data intelligence lifecycle. *Preprint*, arXiv:2512.04324.

Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, Kuan Li, Liangcai Su, Litu Ou, Liwen Zhang, Pengjun Xie, Rui Ye, Wenbiao Yin, Xinmiao Yu, Xinyu Wang, Xixi Wu, and 36 others. 2025a. Tongyi deepresearch technical report. *CoRR*, abs/2510.24701.

Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, Kuan Li, Liangcai Su, Litu Ou, Liwen Zhang, Pengjun Xie, Rui Ye, Wenbiao Yin, Xinmiao Yu, Xinyu Wang, Xixi Wu, and 36 others. 2025b. Tongyi deepresearch technical report. *CoRR*, abs/2510.24701.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhicong Li, Jiahao Wang, Hangyu Mao, ZhiShu Jiang, Zhongxia Chen, Du Jiazhen, Fuzheng Zhang, Di ZHANG, and Yong Liu. 2024. DMQR-RAG: Diverse multi-query rewriting in retrieval-augmented generation.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, and 3 others. 2024. Agentbench: Evaluating llms as agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. 2024. Agentboard: An analytical evaluation board of multi-turn LLM agents. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Guozhao Mo, Wenliang Zhong, Jiawei Chen, Xuanang Chen, Yaojie Lu, Hongyu Lin, Ben He, Xianpei Han, and Le Sun. 2025. Livemcpbench: Can agents navigate an ocean of mcp tools? *Preprint*, arXiv:2508.01780.

Nex-AGI, Yuxuan Cai, Lu Chen, Qiaoling Chen, Yuyang Ding, Liwen Fan, Wenjie Fu, Yufei Gao, Honglin Guo, Pinxue Guo, Zhenhua Han, Zhengfu He, Hanglei Hu, Kai Hu, Shengjia Hua, Tianyu Huai, Baodai Huang, Li Ji, Zhen Jiang, and 46 others. 2025. Nex-n1: Agentic models trained via a unified ecosystem for large-scale environment construction. *Preprint*, arXiv:2512.04987.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. 2025. Training software engineering agents and verifiers with swe-gym. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie S. Kim, Patrick Chao, Samuel Miserendino, Gildas Chabot, David Li, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. 2025. Gdpval: Evaluating AI model performance on real-world economically valuable tasks. *CoRR*, abs/2510.04374.

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, and 81 others. 2025. Humanity's last exam. *CoRR*, abs/2501.14249.

Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. 2024. Adapt: As-needed decomposition and planning with language models. In *Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 4226–4252. Association for Computational Linguistics.

Pranab Sahoo, Prabhash Meharia, Akash Ghosh, Sriparna Saha, Vinija Jain, and Aman Chadha. 2024. A comprehensive survey of hallucination in large language, image, video and audio foundation models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11709–11724, Miami, Florida, USA. Association for Computational Linguistics.

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.

Manan Suri, Puneet Mathur, Nedim Lipka, Franck Dernoncourt, Ryan A. Rossi, and Dinesh Manocha. 2025. Structured uncertainty guided clarification for LLM agents. *CoRR*, abs/2511.08798.

Terminal-Bench. 2025. Terminal-bench: A benchmark for ai agents in terminal environments.

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, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Wenxuan Wang, Juluan Shi, Chaozheng Wang, Cheryl Lee, Youliang Yuan, Jen-tse Huang, and Michael R. Lyu. 2024. Learning to ask: When llms meet unclear instruction. *CoRR*, abs/2409.00557.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, and 2 others. 2025a. Openhands: An open platform for AI software developers as generalist agents. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Shah, Yujia Bao, and Eugene Siow. 2025b. Mcp-bench: Benchmarking tool-using llm agents with complex real-world tasks via mcp servers. *Preprint*, arXiv:2508.20453.

Ziting Wang, Shize Zhang, Haitao Yuan, Jinwei Zhu, Shifu Li, Wei Dong, and Gao Cong. 2025c. Fdabench: A benchmark for data agents on analytical queries over heterogeneous data. *CoRR*, abs/2509.02473.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. Browsecomp: A simple yet challenging benchmark for browsing agents. *CoRR*, abs/2504.12516.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, Songyang Gao, Lu Chen, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. 2025a. Agentgym: Evaluating and training large language model-based agents across diverse environments. 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*, pages 27914–27961. Association for Computational Linguistics.

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, and 4 others. 2025b. Agentgym-rl: Training LLM agents for long-horizon decision making through multi-turn reinforcement learning. *CoRR*, abs/2509.08755.

Frank F. Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Zhiruo Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Maben, Raj Mehta, Wayne Chi, Lawrence Keunho Jang, Yiqing Xie, and 2 others. 2024. Theagentcompany: Benchmarking LLM agents on consequential real world tasks. *CoRR*, abs/2412.14161.

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, and 40 others. 2025a. Qwen3 technical report. *CoRR*, abs/2505.09388.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir

Press. 2024. Swe-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

John Yang, Carlos E. Jimenez, Alex L. Zhang, Kilian Lieret, Joyce Yang, Xindi Wu, Ori Press, Niklas Muennighoff, Gabriel Synnaeve, Karthik R. Narasimhan, Diyi Yang, Sida Wang, and Ofir Press. 2025b. Swe-bench multimodal: Do AI systems generalize to visual software domains? In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

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*.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. 2025. {$\tau$}-bench: A benchmark for \underline{T}ool-\underline{A}gent-\underline{U}ser interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*.

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, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Daoguang Zan, Zhirong Huang, Wei Liu, Hanwu Chen, Linhao Zhang, Shulin Xin, Lu Chen, Qi Liu, Xiaojian Zhong, Aoyan Li, Siyao Liu, Yongsheng Xiao, Liangqiang Chen, Yuyu Zhang, Jing Su, Tianyu Liu, Rui Long, Kai Shen, and Liang Xiang. 2025. Multi-swe-bench: A multilingual benchmark for issue resolving. *CoRR*, abs/2504.02605.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, Kedong Wang, Lucen Zhong, Mingdao Liu, Rui Lu, Shulin Cao, Xiaohan Zhang, Xuancheng Huang, Yao Wei, Yean Cheng, and 80 others. 2025. GLM-4.5: agentic, reasoning, and coding (ARC) foundation models. *CoRR*, abs/2508.06471.

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. *CoRR*, abs/2504.03160.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, Yuxin Gu, Sixin Hong, Jing Ren, Jian Chen, Chao Liu, and Yining Hua. 2025. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *CoRR*, abs/2504.19314.

Yizhang Zhu, Liangwei Wang, Chenyu Yang, Xiaotian Lin, Boyan Li, Wei Zhou, Xinyu Liu, Zhangyang Peng, Tianqi Luo, Yu Li, Chengliang Chai, Chong Chen, Shimin Di, Ju Fan, Ji Sun, Nan Tang, Fugee Tsung, Jiannan Wang, Chenglin Wu, and 6 others. 2025. A survey of data agents: Emerging paradigm or overstated hype? *CoRR*, abs/2510.23587.

## A Usage of AI assistants

We primarily use ai assistants to polish the grammar and wording of the text.

## B Supplement to Benchmark

### B.1 Data Construction

**Complex tool using scenario.** In this scenario, we have a subset of questions that originate from volunteers. We have collected a set of real questions and discussions from 10 volunteers. The volunteers are primarily undergraduate students and above, spanning multiple academic disciplines. During the data collection process, we first introduce the definition of tool discovery to the volunteers—for example, using online services such as WolframAlpha for mathematical formula computation, or accessing official websites to download datasets. We then ask the volunteers to provide recent examples in which they have used tool discovery, along with the corresponding usage tasks. The full instruction is "Please refer to the above definition of tool discovery and, based on your own experience and recent work, provide some scenarios in which tool discovery is used along with the corresponding objectives. Note that these data will be used for benchmark construction, so please do not include any private or sensitive information". From these submissions, we select publicly available data covering a wide range of domains, including academia, daily life, and sports, and conducted subsequent adaptation. These data ultimately form 57 instances for our tool discovery task.

We also involve authors who have a strong understanding of recent deep search work to enhance the existing datasets. The authors are instructed as follows: “Please refer to recent deep search work on the decomposition of entities and relations to break down the existing datasets, and search for or create relevant attachment files to replace key entities, ultimately forming new problems.”

**Data analysis scenario.** In this scenario, 4 authors extensively explore and collect data from GitHub and Kaggle, with a particular focus on datasets related to complex data cleaning. Our authors are instructed as "please use keywords such as data cleaning and data wrangling to search for relevant datasets on websites like GitHub and Kaggle. Priority should be given to complex datasets that come with existing reproduction files and execution traces". We prioritize datasets that provided reproducibility files and conduct manual data inspection and reproduction workflows to ensure the reliability of the entire process. Based on these procedures, we collaborate with 4 volunteers with a background in statistics to create a data-cleaning README for each dataset, requiring "complete, unique, and well-defined cleaning procedures that require sufficient exploration of the dataset itself to be fully understood". We ensure that the cleaning pipeline was unique, while still requiring exploratory data analysis (EDA) of the data itself to complete the process. These data finally form the final 57 instances.

**Deep search scenario.** In this scenario, we collect real question data from social-media users (e.g. Reddit, Xiaohongshu). We prioritize queries that contained hedging adverbs and where the asker had already accepted a correct answer. We manually verify the responses from other users to ensure that only the accepted answer significantly matched the content of the question. The collected data are then standardized—for example, converting "10 years ago" to "before 2015" and removing certain meme emojis. These data ultimately form 48 instances for our Ambiguity-with-bias task.

### B.2 Benchmark statistics

Table 5 details the number and type of attachment files in our benchmark.

| Scenario | File Type | Nums | Total |
|---|---|---|---|
| Complex Tool Usage | csv | 204 | 409 |
| | json | 169 | |
| | jpg | 15 | |
| | png | 9 | |
| | mp4 | 5 | |
| | txt | 3 | |
| | pdf | 2 | |
| | svg | 1 | |
| | py | 1 | |
| Data Analysis | csv | 181 | 243 |
| | md | 26 | |
| | json | 20 | |
| | txt | 6 | |
| | py | 3 | |
| | yml | 2 | |
| | pdf | 1 | |
| | tsv | 1 | |
| | html | 1 | |
| | yaml | 1 | |
| | log | 1 | |
| Deep Search | — | — | — |
| Total | — | — | 652 |

Table 5: Distribution of attachment file types across different scenarios on AgentGym2.

## C Details on toolbox

The detailed definition of 27 tools across web browsing, information retrieval, file processing, multimodal understanding and code execution are shown in Table 6. In practice, all tools are encapsulated as functions, and their definitions along with the input–output formats of parameters are passed to the agent via the OpenAI tool schema.

## D Experimental Setup Details

### D.1 LLM Implementation Details

For open-source models under 100B parameters, we deploy them locally on NVIDIA A100 GPUs and provide an OpenAI-compatible server using the vllm library(Kwon et al., 2023). For other models, we access the official providers and conduct standardized testing through their official APIs.

### D.2 Other Implementation Details

To prevent potential data contamination, we manually disable the models' access to the relevant websites when testing queries sourced from real online data.

## E Prompt Details

### E.1 System Prompt

The system prompt details, adapted from owl (Hu et al., 2025), for our evaluation is shown in Figure 10. To make a fair comparison between different models, we use this same prompt to every query and to every model. This system prompt, together with tool definitions, are prompted to the agent first before the task query.

### E.2 LLM-as-Judge Prompt

The LLM-as-Judge prompt specifications are illustrated in Figure 11, Figure 12, and Figure 13. To ensure reliable evaluation performance, we adopt three scenario-specific prompts, adapted from Tongyi-DeepResearch(Li et al., 2025b).

## F Failure Case Study

### F.1 Failure mode analysis

To further investigate the models' failure modes, we manually examine and summarize several types of errors. We use GPT-5 to annotate the error types for incorrect data. We provide a possible correct reference answer trajectory, along with our definitions for each error type, as shown in Table 7.

### F.2 Failure Cases

We provide three detailed case studies—one per scenario—to illustrate these failure modes. In deep search, the agent fails to verify information under memory bias and, after multiple rounds, becomes convinced that an initially incorrect answer satisfies all constraints (Figures 14–16). In complex tool use, the agent under-explores the course webpage tool, makes unsupported inferences about the assignment files, and ultimately downloads the wrong file without direct evidence (Figures 17–19). In data analysis, the agent does not inspect the dataset's properties, misinterprets the task requirements, and this error propagates to the subsequent analysis and final answers (Figures 20–21).

## G Justification about metric reliability

We validate our metric reliability in the Deep Search scenario by sampling each query 8 times. We then estimate uncertainty intervals for Pass@3 and Avg@3 via resampling: for each metric, we compute scores from all $C_8^3$ possible 3-sample subsets of the 8 runs and average the results. The Table 8 reports the results.

The results show that the uncertainty intervals are all narrow, and the relative ranking remains consistent. This indicates that under a budget of 3 samples, the metrics are not highly sensitive to sampling randomness, supporting the reliability of our chosen main metrics.

**System Prompt**

===== RULES OF ASSISTANT =====
You are a helpful assistant. You have to utilize your available tools to solve the task I assigned. Please note that our overall task may be very complicated. Here are some tips that may help you solve the task:
<tips>
- If one way fails to provide an answer, try other ways or methods. The answer does exists.
- If the search snippet is unhelpful but the URL comes from an authoritative source, try visit the website for more details.
- When looking for specific numerical values (e.g., dollar amounts), prioritize reliable sources and avoid relying only on search snippets.
- When solving tasks that require web searches, check Wikipedia first before exploring other websites.
- When trying to solve math problems, you can try to write python code and use sympy library to solve the problem.
- Always verify the accuracy of your final answers!
- Do not be overly confident in your own knowledge.
- You can use browser tools to access full browser functionality, including rich interactions, file downloads, and page visits, etc.
- After writing codes, do not forget to run the code and get the result.
- When a tool fails to run, never assume that it returns the correct result.
- Do not attempt to read or parse any files directly.
- Always read all README files before taking action.
</tips>
Here are some hint about the final answer after solving the whole task:
<hint>
Your final answer must be enclosed by <final_answer></final_answer>.
Your final answer must be output exactly in the format specified by the question.
</hint>
Here is the overall task:
{task}
Please solve this task step by step. Never forget the task!

Figure 10: System Prompt of AgentGym2.

| Name | Description | Input | Output |
|---|---|---|---|
| ***- Web Browsing*** | | | |
| scrape | Fetch and extract the main textual content from a given webpage. | [text] URL. [text] Query | [text] Extracted webpage content |
| browse_url | A powerful toolkit which can simulate the browser interaction to solve the task which needs multi-step actions. | [text] Task prompt. [text] Start URL. [int] Round limit | [text] Task completion result |
| ***- Supported Browser Actions (used within*** `browse_url`***)*** | | | |
| fill_input_id | Fill an input field and submit the text. | [text/int] Element ID. [text] Input text | Page updated |
| click_id | Click an element on the page. | [text/int] Element ID | Page updated |
| hover_id | Hover over an element to trigger UI changes. | [text/int] Element ID | Page updated |
| download_file_id | Download a file from the page. | [text/int] Element ID | [file] Local file path |
| scroll_to_top/bottom | Scroll to the top/bottom of the page. | — | Page updated |
| scroll_up/down | Scroll upward/downward within the page. | — | Page updated |
| back | Navigate back to the previous page. | — | Page updated |
| stop | Stop the browsing process and finalize the result. | — | [text] Final result |
| get_url | Retrieve the current page URL. | — | [text] Current URL |
| find_text_on_page | Find and scroll to specific text on the page. | [text] Search text | Page scrolled |
| visit_page | Navigate directly to a specified URL. | [text] URL | Page updated |
| click_blank_area | Click a blank area to remove focus from active elements. | — | Page updated |
| ask_question_about_video | Ask a question about video content on the page. | [text] Question | [text] Answer |
| ***- Information Retrieval*** | | | |
| search_wiki | Search Wikipedia for a specific entity and return its factual summary. | [text] Entity name | [text] Entity summary |
| search_serper | Perform Google web or news search and return top search results. | [text] Query and parameters | [text] Search results |
| google_scholar | Retrieve relevant academic publications from Google Scholar. | [text] Academic query | [text] Scholarly results |
| ***- File Processing*** | | | |
| extract_document_content | Extract the content of a document and return the processed text. | [file] Document path | [text] Extracted text |
| extract_excel_content | Extract detailed cell and sheet information from an Excel file. | [file] Excel path | [text] Structured content |
| ***- Multimodal Understanding*** | | | |
| audio2text | Transcribe audio content into text. | [audio] File or URL | [text] Transcription |
| ask_question_about_audio | Answer questions based on the semantic understanding of audio. | [audio] File. [text] Question | [text] Answer |
| image_to_text | Generate a textual description of an image. | [image] File. [text] Prompt | [text] Description |
| ask_question_about_image | Answer questions about image content with optional instructions. | [image] Image file or URL. [text] Question. [text] System prompt | [text] Answer |
| ***- Code Execution*** | | | |
| execute_code | Execute a code snippet in a stateless environment and return the result. | [text] Source code | [text] Execution result |

Table 6: Detailed definition and prompts of tools across categories on AgentGym2.

**LLM-as-Judge Prompt for Complex Tool Usage**

Based on the given question, standard answer, and model-predicted answer, evaluate whether the model's response is correct. Your task is to classify the result as: [CORRECT] or [INCORRECT]. First, we'll list examples for each category, then you'll evaluate a new question's predicted answer.
Here are examples of [CORRECT] responses:
Question: What are the names of Barack Obama's children?
Standard Answer: Malia Obama and Sasha Obama
Model Prediction 1: Malia Obama and Sasha Obama
Model Prediction 2: Malia and Sasha
Model Prediction 3: Most would say Malia and Sasha, but I'm not sure, I should verify
Model Prediction 4: Barack Obama has two daughters, Malia Ann and Natasha Marian, commonly known as Malia Obama and Sasha Obama.
These responses are all [CORRECT] because they:
- Fully include the important information from the standard answer.
- Don't contain any information that contradicts the standard answer.
- Focus only on semantic content; language, capitalization, punctuation, grammar, and order aren't important.
- Vague statements or guesses are acceptable as long as they include the standard answer and don't contain incorrect information or contradictions.
Here are examples of [INCORRECT] responses:
Question: What are the names of Barack Obama's children?
Standard Answer: Malia Obama and Sasha Obama
Model Prediction 1: Malia
Model Prediction 2: Malia, Sasha and Susan or Sasha Obama or Malia Obama, or Natasha Marian, or Einstein
Model Prediction 3: While I don't know their exact names, I can tell you Barack Obama has two children.
Model Prediction 4: You might be thinking of Betsy and Olivia. But you should verify the details with the latest references. Is that the correct answer?
Model Prediction 5: Barack Obama's children
These responses are all [INCORRECT] because they:
- Contain factual statements that contradict the standard answer.
- Are empty or merely repeat the question.
- Enumerate multiple answers or repeat the answer.

Below is a new question example. Please reply with only [CORRECT] or [INCORRECT], without apologies or corrections to your own errors, just evaluate the answer.
Question: {question}
Standard Answer: {correct_answer}
Predicted Answer: {response}
Evaluate this new question's predicted answer as one of the following:
A. [CORRECT]
B. [INCORRECT]
Return only the option representing [CORRECT] or [INCORRECT], i.e., just return A or B, without adding any other text.

Figure 11: LLM-as-Judge prompt on Complex Tool Usage task.

**LLM-as-Judge Prompt for Data Analysis**

You are an evaluation assistant. Please determine if the predicted answer is equivalent to the labeled answer.
Question: {question}
Labeled Answer: {correct_answer}
Predicted Answer: {response}
Did the model give an answer **equivalent** to the labeled answer? Please respond with "Correct" if they are equivalent, or "Incorrect" if they are not equivalent. Do not include any other text.

Figure 12: LLM-as-Judge prompt on Data Analysis task.

| **Error Type** | **Definition** |
|---|---|
| Instruction Misinterpretation | The model fails to understand the problem, does not follow the instructions in the prompt, and does not produce the correct answer format—for example, not reading the README.md. |
| Insufficient Exploration | The model fails to sufficiently explore the environment (e.g., discovering necessary tools, inspecting datasets, or checking relevant links), resulting in a lack of essential clues. |
| Incorrect Analysis | The model has already obtained the correct facts, data, or clues, but makes mistakes when reasoning or analyzing based on them. |
| Guessing or Fabrication | The model is unable to continue its reasoning process and instead guesses a clue or an answer based on its internal knowledge or previously unrelated content, then proceeds with the reasoning or response. |
| Confirmation Bias | After obtaining relevant clues or an answer, the model fails to sufficiently verify them and instead directly assumes they are correct, even forcing other clues to be interpreted to fit these incorrect conclusions. |
| Tool Execution Failure | The model repeatedly encounters errors when calling tools, possibly due to incorrect parameter settings, malformed tool-call formats, or internal issues within the tools. |
| Over-length | The model's trajectory exceeds the token length limit or the maximum number of interaction turns. |
| Others | Other errors that do not fall into the failure modes listed above. |

Table 7: Failure modes description used in our error analysis.

**LLM-as-Judge Prompt for Deep Search**

Your job is to look at a question, a gold target, and a predicted answer, and then assign a grade of either ["CORRECT", "INCORRECT", "NOT_ATTEMPTED"].
First, I will give examples of each grade, and then you will grade a new example.
The following are examples of CORRECT predicted answers.
Question: What are the names of Barack Obama's children?
Gold target: Malia Obama and Sasha Obama
Predicted answer 1: sasha and malia obama
Predicted answer 2: most people would say Malia and Sasha, but I'm not sure and would have to double check.
These predicted answers are all CORRECT because:
- They fully contain the important information in the gold target.
- They do not contain any information that contradicts the gold target.
- Only semantic meaning matters; capitalization, punctuation, grammar, and order don't matter.
- Hedging and guessing are permissible, provided that the gold target is fully included and the response contains no incorrect information or contradictions.
The following are examples of INCORRECT predicted answers.
Question: What are the names of Barack Obama's children?
Gold target: Malia and Sasha
Predicted answer 1: Malia.
Predicted answer 2: Malia, Sasha, and Susan.
Predicted answer 3: Barack Obama does not have any children.
Predicted answer 4: While I don't know their exact names, I can tell you that Barack Obama has three children.
These predicted answers are all INCORRECT because:
- A factual statement in the answer contradicts the gold target. Incorrect statements that have some hedging (e.g., "it is possible that", "although i'm not sure, i think") are also considered incorrect.
The following are examples of NOT_ATTEMPTED predicted answers.
Question: What are the names of Barack Obama's children?
Gold target: Malia and Sasha
Predicted answer 1: I don't know.
Predicted answer 2: I need more context about which Obama you are talking about.
Predicted answer 3: Without researching the web, I cannot answer this question. However, I can tell you that Barack Obama has two children.
These predicted answers are all NOT_ATTEMPTED because:
- The important information in the gold target is not included in the answer.
- No statements in the answer contradict the gold target.
Here is a new example. Simply reply with either CORRECT, INCORRECT, NOT ATTEMPTED. Don't apologize or correct yourself if there was a mistake; we are just trying to grade the answer.
Question: {question}
Gold target: {correct_answer}
Predicted answer: {response}
Grade the predicted answer of this new question as one of:
A: CORRECT
B: INCORRECT
C: NOT_ATTEMPTED
Just return the letters "A", "B", or "C", with no text around it.

Figure 13: LLM-as-Judge prompt on Deep Search task.

A failure case in deep search scenario.

**User:**

Looking for an animated movie I watched as a child; details might be misremembered. The movie had a hand-drawn animation style from the 90s to 2000s era, not computer-generated. The visuals were dark, somewhat like old Disney animations but not quite the same. The story setting might be about fishing or an Arctic story. There was one particularly scary scene where the protagonists—possibly penguins or seals—were fishing when they got chased by **a black submarine or some kind of vessel with a huge mouth**. The submarine's mouth opened and closed, devouring fish. When the submarine opened its mouth to swallow fish, the mouth would emit an eerie red-orange glow. Apart from this scene and the Arctic setting, I can't remember much else. One of the movie's themes might be overfishing. What is the English name of this movie?

**Assistant:**

Let me search more specifically for animated movies about overfishing or Arctic themes:

**Action: Search Serper**

```
<tool_call>
{
  "function_name": "search_serper",
  "parameters": {
     "query": "animated film \"overfishing\" Arctic penguins seals 1990s 2000s",
  }
}
</tool_call>
```

**Tool Call/Response**


```
[
  {
    'title': Happy Feet (2006) - Plot',
    'link': 'https://www.imdb.com/title/tt0366548/plotsummary/',
    'snippet': ... Antarctic overfishing, satisfying both the emperor penguins
and the Amigos. Contribute to this page. Suggest an edit or add missing
content. Learn more about ... Read more',
     'position': 1
  },
  {
    'title': 'Happy Feet | Warner Bros. Entertainment Wiki - Fandom',
    'link': 'https://warnerbros.fandom.com/wiki/Happy_Feet',
    'snippet': 'The governments realize they are overfishing, leading to the
banning of all Antarctic fishing. At this, the Emperor Penguins and the Amigos
celebrate. In the ... Read more',
    'position': 2
  }
]
```


Figure 14: Part 1/3 of the deep search case study: The text in red in the question represents the user's memory bias. The model first combines and searches for information based on the keywords in the question.

A failure case in deep search scenario.

**Assistant:**

Interesting! The search results are pointing to "Happy Feet" (2006), which is about penguins and overfishing. However, I don't recall a submarine with a glowing mouth in Happy Feet. Let me search more specifically for the submarine scene:

**Action: Search Serper**

```
<tool_call>
{
  “function_name”: “search_serper”,
  “parameters”: {
     “query”: “\"Happy Feet\" submarine scene fishing ship mouth glowing”,
  }
}
</tool_call>
```

**Tool Call/Response**


```
[
  {
    'title': 'Best Orca Scene from Happy Feet Explained’,
    'link': 'https://www.tiktok.com/@orcas4life/video/7491303189778648342’,
    'snippet': 'Explore the memorable orca moments in Happy Feet and why they
stand out as the coolest part of the film. #orca #killerwhale #happyfeet
#animal.’,
     'position’: 1
  },
  {
     'Arctic Encounters: an orcaholics expedition in Norway’,
     'link': 'https://explorearth.com/original-stories/story/arctic-
encounters-an-orcaholics-expedition-in-norway/’,
     'snippet': ' ... Happy Feet\\'. These CGI orcas were referred to as
\"demon fish\" by the late Robin Williams penguin character Lovelace. Orcas
have even appeared in some ... '
     'position’: 2
  }
]
```


**Assistant:**

Let me search for other animated films with submarines or fishing themes from that era:

... continued on the next page)

Figure 15: Part 2/3 of the deep search case study: The model finds an initial answer and intends to verify its information. However, due to **Insufficient Exploration**, it fails to perform proper verification and cannot identify other possible answers.

A failure case in deep search scenario.

**Assistant:**

Based on your detailed description of an animated film featuring:- 1990s-2000s hand-drawn animation style - Dark, gloomy atmosphere - Arctic/Antarctic setting with seals or penguins - A terrifying scene with a black submarine-like vessel with a mouth that opens and closes - The submarine's mouth glowing red-orange when eating fish - Theme of overfishing
And considering the search results, I believe the film you're looking for is **Happy Feet** (2006). However, let me verify this by searching for more specific scenes in Happy Feet that might match your description of the submarine with a glowing mouth:

**Action: Search Serper**

```
<tool_call>
{
  “function_name”: “search_serper”,
  “parameters”: {
     “query”: “\"Happy Feet\" submarine scene fishing ship mouth glowing”,
  }
}
</tool_call>
```

**Tool Call/Response**

```
[
  {
    'title': 'Movie: Happy Feet - Learn to Dive Today’,
    'link': 'https://www.learntodivetoday.co.za/blog/2012/09/08/movie-happy-feet/’,
    'snippet': 'We watched Happy Feet during a time of stress (preparing to move house!), and found it (mostly) calming and funny.’,
     'position’: 1
  },
]
```

**Assistant:**

Based on my extensive research, the animated film you're describing with the specific elements you mentioned (penguins/seals, Arctic setting, dark atmosphere, overfishing theme, and a terrifying submarine/fishing vessel with a glowing mouth) appears to be **Happy Feet** (2006).
```
<final_answer>
Happy Feet
</final_answer>
```

Figure 16: Part 3/3 of the deep search case study: After failing to find other possible answers based on the clues, the model revisits and tries to further verify the initial answer. However, due to imprecise keywords, it still cannot confirm whether the answer is correct or incorrect. Ultimately, the model falls into **Confirmation Bias** and directly outputs this incorrect initial answer.

A failure case in complex tool use scenario.

**User:**

Design and Analysis of Algorithms is an important course, offered by many universities. Please help me find the course as taught at the university ranked 3rd in the **2026 QS rankings.** This course is offered in many quarters, and in one quarter, the first class is held on a Monday in January. Please help me solve problem 3.2 from the assignment **published on Mar 1 in that quarter**.

**Assistant:**

I'll help you solve this task step by step. Let me start by finding the university ranked 3rd in the 2026 QS rankings.

**Action: Scrape**

```
<tool_call>
{
  "function_name": "scrape",
  "parameters": {
     "query": "top 10 universities 2026 QS rankings",
     "url": "https://www.qs.com/insights/qs-world-university-rankings",
  }
}
</tool_call>
```

**Tool Call/Response**

```
Top 10 Universities (QS World University Rankings 2026):
1st: Massachusetts Institute of Technology (MIT), US
2nd: Imperial College London, UK
3rd: Stanford University, US
4th: University of Oxford, UK
5th: Harvard University, US
6th: University of Cambridge, UK
7th: ETH Zurich, Switzerland
8th: National University of Singapore (NUS), Singapore
9th: University College London (UCL), UK
10th: California Institute of Technology (Caltech)
```

Figure 17: Part 1/3 of the complex tool use case study part 1/3: The task requires the model to use the QS Rankings tool and the corresponding course webpage tool to download files. The model first correctly uses the QS Rankings tool through the basic tools to obtain information.

A failure case in complex tool use scenario.

**Assistant:**

Excellent! I found the Winter 2025 CS161 course. Let me visit the Winter 2025 sections page to find the schedule and determine when the first class was held.

**Action: Scrape**

```
<tool_call>
{
  "function_name": "scrape",
  "parameters": {
     "query": "lecture schedule Monday January first class",
     "url": "https:// stanford-cs161.github.io/winter2025/",
  }
}
</tool_call>
```

**Tool Call/Response**

```
Lecture Schedule - Monday January First Class (Stanford CS161, Winter 2025)
1. Course Overview:
* Instructors: Nima Anari and Moses Charikar
* Lecture Days/Times: Monday and Wednesday, 10:30 AM - 12:00 PM
* Location: Skilling Auditorium
* Start Date: Monday, January 6, 2025 (assumed based on typical Stanford winter
quarter start)
2. Course Content:
* Topics include algorithm analysis, data structures (BSTs, heaps, hash tables),
divide-and-conquer, dynamic programming, greedy algorithms, graph algorithms, and
possible coverage of network flow and string searching.
3. Assessment Structure:
* 8 homework assignments (lowest dropped, each worth approximately 7.143% of the
grade)
* Midterm exam (20%)
* Final exam (30%)
Relevant Links:
* Ed Discussion: For course announcements and staff communication (private or
public posts)
* Email Contact: [cs161-staff-win2425@cs.stanford.edu](mailto:cs161-staff-
win2425@cs.stanford.edu) - For sensitive or personal inquiries
Note:
This schedule aligns with a Monday first class in January for Stanford's Winter
2025 quarter.
```

**Assistant:**

Perfect! I found that Stanford's CS161 Winter 2025 course has its first class on **Monday, January 6, 2025** at 10:30 AM in Skilling Auditorium. The instructors are Nima Anari and Moses Charikar.

Now I need to find the assignment **published on March 1 in that quarter** and solve problem 3.2. Let me search for the assignments.

(... continued on the next page)

Figure 18: Part 2/3 of the complex tool use case study: The model then intends to locate and verify the correct assignment files. However, due to **Incorrect Analysis**, although it finds the files and publication dates for homework1 and homework2, it incorrectly infers the publication date for homework5.

A failure case in complex tool use scenario.

**Assistant:**

Let me try a different approach and search for specific homework files on the Stanford CS161 website.

**Action: Search Serper**

```
<tool_call>
{
  "function_name": "search_serper",
  "parameters": {
     "query": "site:stanford-cs161.github.io/winter2025 homework1 OR
homework2 OR homework3 OR homework4 OR homework5 OR homework6 OR homework7",
  }
}
</tool_call>
```

**Tool Call/Response**

```
[
  {
    'title': 'Schedule - CS 161',
    'link': 'https://stanford-cs161.github.io/winter2025/schedule/',
    'snippet': 'Lecture 2: Asymptotics, Worst-Case Analysis, and
MergeSort · Homework 1; Thu, Jan 9 ; Section 1 · Section 1 · Fri, Jan 10 ;
Homework 1 · Homework 2; Thu, Jan 16 ... See more',
     'position': 1
  },
]
```

**Assistant:**

Perfect! I found problem 3.2 in Homework 5. The problem asks:
3.2 Expected vs. deterministic (1pt.) Why does an expected O(1) runtime for SEARCH not imply a deterministic worst-case O(1) runtime? [We are expecting: A brief explanation in plain English.]
```
<final_answer>
Expected O(1) runtime averages over random hash function choices, but worst-case O(1) must hold for every specific hash function. A particular hash function may create many collisions for certain key sets, resulting in worst-case O(n) search time even when the expectation over all hash functions is O(1).
</final_answer>
```

Figure 19: Part 3/3 of the complex tool use case study: Due to **Insufficient Exploration** of the course webpage tool, the model never realizes that it selected the wrong semester and ultimately falls into **Confirmation Bias**, downloading the wrong file.

A failure case in data analysis scenario.

**User:**

Please process the data files according to the method described in data cleaning process and answer the following question.
Question: In data frame country_doctors_total_avg_filtered, how many unique countries are there and how many rows of data are there?

**Assistant:**

Let me read the README file using code

**Action: Code**

```
<tool_call>
{
  "function_name": "execute_code",
  "parameters": {
     "code": "import pandas as pd\n\n# Read the README file\nwith
open('/mnt/data/mai_6/35b6dfed-d20b-5473-b372-27bef031c391/README.md', 'r')
as f:\n    readme_content = f.read()\n\nprint(readme_content)",
  }
}
</tool_call>
```

**Tool Call/Response**

```
Executed Results:
…
- For the number of Medical Doctors, some countries were later filtered out
from the original data set because of non-inferable computationally or missing
values.
…
#### Immunization Rates
The purpose of this analysis was to analyze the average immunization rates of
150 countries and determine if there were any correlations between average
immunization rates and medical doctor prevalence, and average immunization
rates and life expectancy.
```

Figure 20: Part 1/2 of the data analysis case study: The task requires the model to perform the corresponding cleaning and analysis based on the problem background and cleaning rules provided in the README.

A failure case in complex tool use scenario.

**Assistant:**

Let me run this in a single execution to maintain variable continuity:

**Action: Code**

```
<tool_call>
{
  "function_name": "execute_code",
  "parameters": {
     "code": "…Step 4: Create country_doctors_fm_filtered (filter out null
medical doctors)\ncountry_doctors_fm_filtered =
country_doctors_fm[country_doctors_fm['Medical doctors
(number)'].notnull()].copy()",
  }
}
</tool_call>
```

**Tool Call/Response**

```
Executed Results:
country_doctors_fm_filtered shape: (394, 23)
Unique countries in fm_filtered: 122
country_doctors_total shape: (0, 23)
Unique countries in total: 0
Medical doctors (number) null count: 0
```

Figure 21: Part 2/2 of the data analysis case study: However, due to the model's **Insufficient Exploration**, it has limited understanding of the dataset itself, leading to a misinterpretation of the task during cleaning and the direct removal of all null data. This **Instruction Misinterpretation** directly results in the final incorrect answer.

| Models | Pass@3 | Avg@3 |
|---|---|---|
| GPT-5 | 65.4±1.5 | 48.2±1.1 |
| Gemini2.5-Pro | 41.6±2.3 | 26.2±1.7 |
| Nex-N1-671B | 53.1±2.1 | 34.3±1.8 |
| Kimi-K2 | 38.7±2.0 | 21.9±1.7 |

Table 8: Metric reliability under more sampling.