MiroThinker: Pushing the Performance Boundaries of Open-Source Research Agents via Model, Context, and Interactive Scaling

MiroMind Team

We present MiroThinker v1.0, an open-source research agent designed to advance tool-augmented reasoning and information-seeking capabilities. Unlike previous agents that only scale up model size or context length, MiroThinker explores interaction scaling at the model level—systematically training the model to handle deeper and more frequent agent–environment interactions as a third dimension of performance improvement. Unlike LLM test-time scaling, which operates in isolation and risks degradation with longer reasoning chains, interactive scaling leverages environment feedback and external information acquisition to correct errors and refine trajectories. Through reinforcement learning, the model achieves efficient interaction scaling: with a 256K context window, it can perform up to 600 tool calls per task, enabling sustained multi-turn reasoning and complex real-world research workflows. Across four representative benchmarks—GAIA, HLE, BrowseComp, and BrowseComp-ZH—the 72B variant achieves up to 81.9%, 37.7%, 47.1%, and 55.6% accuracy respectively, surpassing previous open-source agents and approaching commercial counterparts such as GPT-5-high. Our analysis reveals that MiroThinker benefits from interactive scaling consistently: research performance improves predictably as the model engages in deeper and more frequent agent–environment interactions, demonstrating that interaction depth exhibits scaling behaviors analogous to model size and context length. These findings establish interaction scaling as a third critical dimension for building next-generation open research agents, complementing model capacity and context windows.

Online Demo: https://dr.miromind.ai
Code Repository: https://github.com/MiroMindAI/MiroThinker
Model Weights: https://huggingface.co/miromind-ai/MiroThinker-v1.0-72B

1. Introduction

The rapid evolution of Large Language Models (LLMs) has sparked a paradigm shift in artificial intelligence, from static text generators to dynamic, tool-augmented agents capable of reasoning and interacting with the real world [1–7]. Within this emerging paradigm, research capability has become a new frontier of intelligence. Achieving research-level reasoning requires more than linguistic fluency; it demands the ability to formulate hypotheses, retrieve and verify evidence, and synthesize insights across diverse information sources. Proprietary systems such as ChatGPT Agent [8] and Claude Research [9] exemplify the potential of such capabilities, demonstrating near-human proficiency in literature review, comparative analysis, and reasoning-driven knowledge discovery. However, these systems remain closed, constraining transparency, reproducibility, and community-driven innovation.

To narrow the gap with proprietary systems, the open-source community has made remarkable progress in recent years. At the foundation model level, open-weight LLMs are increasingly trained with built-in agentic skills such as search, browsing, and coding [2–5, 10]. Yet these projects typically release only model weights without providing the full suite of tools or agentic frameworks necessary for end-to-end research reasoning. Meanwhile, a parallel line of open-source work [11–22] focuses on developing specialized research agent models, along with their corresponding toolchains and frameworks. However, these models are often

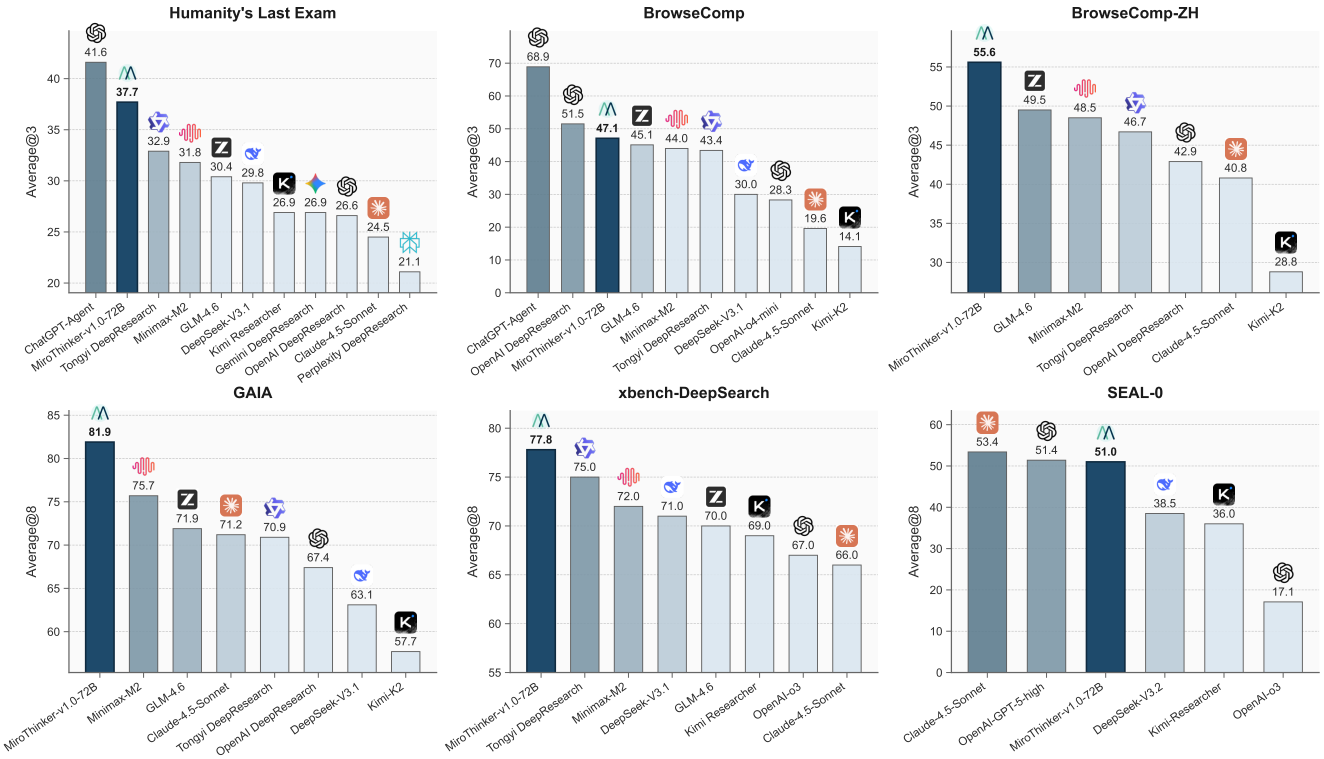


Figure 1: Comparison of MiroThinker with state-of-the-art agents and agentic foundation models.

relatively small in scale and constrained in context length and interaction depth, leaving a clear performance gap compared with leading commercial research agents [8, 9, 23, 24].

In response to these challenges, we introduce `MiroThinker v1.0`, an open-source, high-performance research agent model that pushes the performance boundaries of open-source systems along three key dimensions: *model size*, *context length*, and *interaction depth*. Leveraging these capabilities, the model engages in iterative cycles of reasoning and tool use, enabling it to decompose complex research problems, retrieve and integrate real-time information from the Internet, synthesize multi-source evidence, and generate transparent, well-grounded conclusions. **To sustain such deep reasoning processes, the model is equipped with a 256K context window, supporting up to 600 tool calls per task, a significant leap from the previous open-source models of fewer than 100.** To support diverse computational budgets, we release MiroThinker v1.0 in 8B, 30B, and 72B variants, together with a comprehensive suite of tools.

As shown in Figure 1, empirical evaluations demonstrate that MiroThinker v1.0, equipped with a simple ReAct agent, achieves state-of-the-art performance among open-source research agents, approaching the results of leading commercial systems [1, 8, 9, 23, 24]. Specifically, on the BrowseComp [25] benchmark, MiroThinker-v1.0-72B achieves 47.1% accuracy, surpassing MiniMax-M2 [3] by 2.0 points. A similar pattern is observed on the BrowseComp-ZH [26] benchmark, with a 6.1-point improvement over GLM-4.6 [4], underscoring the model's robust multilingual reasoning ability. Further, on Humanity's Last Exam (HLE) [27], MiroThinker-v1.0-72B continues to outperform its peers, achieving a score of 37.7%, 4.8 points higher than Tongyi-DeepResearch [11]. On the GAIA-Text-Only benchmark [28], MiroThinker-v1.0-72B attains a score of 81.9%, surpassing its strongest open-source counterpart MiniMax-M2 [3] (75.7%) by 6.2 points. Overall, across diverse benchmarks, MiroThinker v1.0 maintains a consistent and substantial advantage, demonstrating stronger reasoning capability, long-context comprehension, and deep tool-use proficiency than existing open-source research agents.

2. Related Works

Agent Foundation Models Recent research has increasingly emphasized enhancing Large Language Models (LLMs) with agentic capabilities—the ability to plan, reason, and act autonomously in complex environments [1–7, 10]. Building on this trend, Agent Foundation Models (AFMs) have emerged as a new paradigm of foundation models that, beyond learning general language understanding, explicitly incorporate agent-oriented abilities, such as decision-making, tool use, and interaction with external environments, during their base model training. Current efforts particularly focus on code and search agents, aiming to enhance capabilities in tool-based problem solving, retrieval-augmented reasoning, and autonomous task execution. Representative AFMs such as GPT-5 [1], Claude-4.5 [7], Grok-3 [29], Kimi K2 [2], MiniMax M2 [3], GLM-4.6 [4], and DeepSeek-V3.1 [5] have demonstrated promising progress in these domains, underscoring a shift toward more specialized and practical forms of agentic intelligence.

Deep Research Models Continuing this progression, deep research models have been introduced as specialized LLM-based agents for complex multi-hop reasoning and long-context, retrieval-intensive tasks. These models integrate dynamic information-seeking and iterative planning into their workflows, enabling autonomous acquisition and synthesis of knowledge into comprehensive answers. Major AI labs have accordingly developed proprietary deep research systems: OpenAI Deep Research [24], Claude Research [9], Kimi-Researcher [23], Grok DeepSearch [29], *etc.*, all extend LLMs with agentic tool use and long-horizon reasoning tailored for in-depth research. Meanwhile, the open-source community has introduced many deep research models. For example, WebThinker [12], WebSailor [13], WebShaper [14], and Tongyi DeepResearch [11] leverage LLMs with iterative web-browsing workflows, whereas Cognitive Kernel-Pro [15], AFM [16], WebDancer [18], and DeepMiner [17], *etc.*, explore novel training algorithms and dynamic memory mechanisms to push the boundaries of research capabilities. Collectively, these developments underscore a broader shift toward LLMs that serve as specialized research assistants, combining advanced reasoning with real-time information retrieval to tackle open-ended knowledge-intensive tasks.

3. Agentic Workflow

3.1. Formulation

MiroThinker v1.0 model is developed under the ReAct paradigm [30] in a single-agent setting. Given a query q, the model alternates between reasoning, tool invocation, and observation in an iterative loop until termination. At step t, the agent maintains a trajectory

$$H_t = \{(T_1, A_1, O_1), \dots, (T_{t-1}, A_{t-1}, O_{t-1})\}, \tag{1}$$

where T_i, A_i, and O_i denote the thought, action, and observation, respectively. The thinking model f_θ generates an internal thought context $T_t = f_\theta(q, H_t)$, followed by an action policy

$$A_t = \pi_\theta(H_t, T_t), \tag{2}$$

which takes as input the trajectory history H_t and the current thought T_t, and outputs a structured tool invocation that specifies which external tool to use and how to query it. The environment executes the invocation and returns a tool response $O_t = \text{Tool}(A_t)$, which is appended to form

$$H_{t+1} = H_t \cup \{(T_t, A_t, O_t)\}. \tag{3}$$

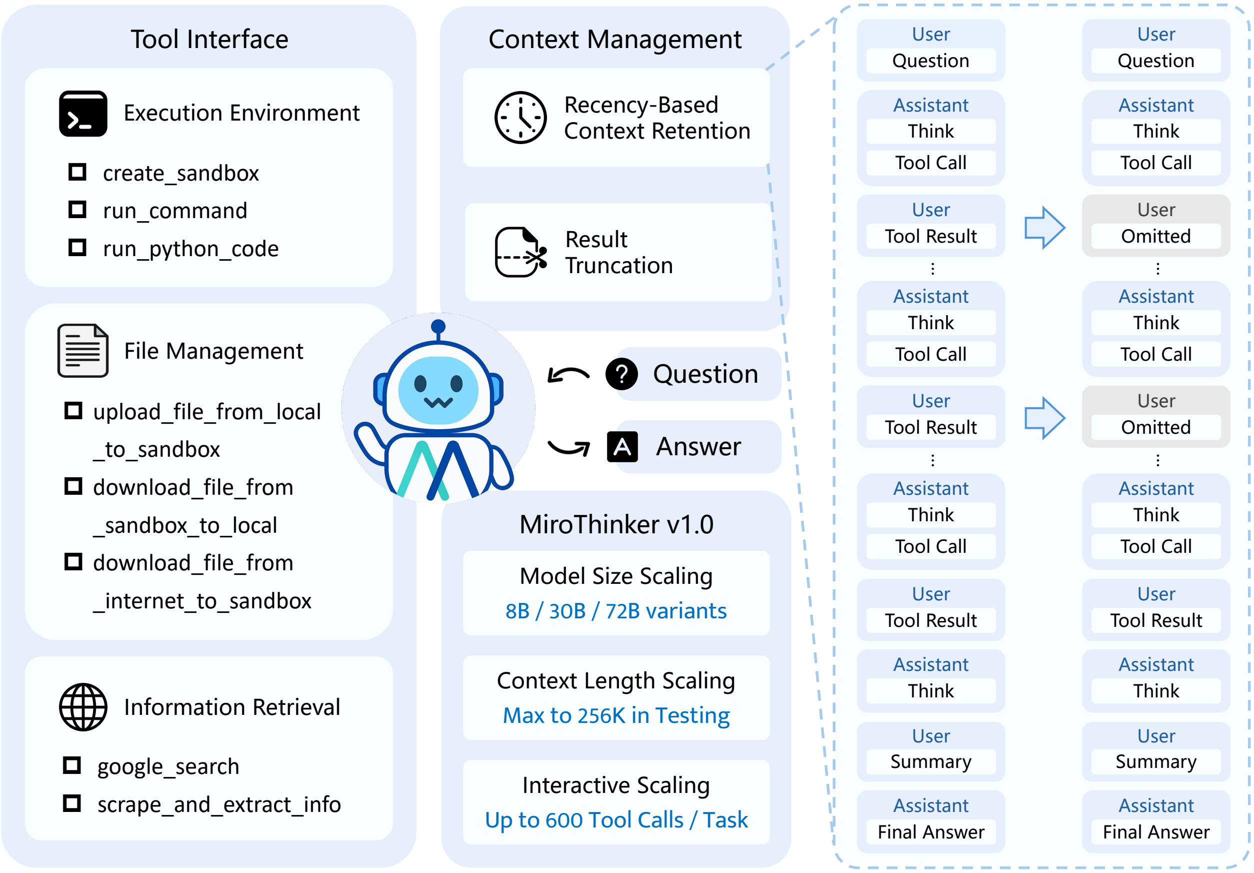


Figure 2: Overview of the MiroThinker v1.0 agent architecture. The framework integrates a structured tool interface, *i.e.,* execution environment, file management, and information retrieval, with a simple recency-aware context management to support interactive scaling. On the right, an agentic trajectory example illustrates the recency-based context retention mechanism, where tool outputs from earlier turns are omitted to maintain context efficiency.

This reasoning–acting–observing loop continues until the model outputs no further action ($A_t = \varnothing$), upon which a summary phase produces the final answer $y = g_\theta(H_t)$. This iterative workflow enables dynamic reasoning grounded in external evidence, yielding interpretable and adaptive decision-making compared with static single-pass LLMs.

3.2. Tool Interface

To enable interaction with external environments, the model is equipped with a modular tool interface that exposes a set of tools. Each tool encapsulates a specific capability (*e.g.*, code execution, file handling, or web retrieval), allowing the model to act beyond pure text generation.

Execution Environment We employ a Linux sandbox that provides an isolated runtime for command and code execution. The agent can create a sandbox instance via `create_sandbox` and subsequently execute shell commands (`run_command`) or Python code (`run_python_code`) within it. This design ensures safe and flexible interaction with system-level resources.

File Management To facilitate data flow between the sandbox and the external world, we implement file upload and download utilities. Specifically, `upload_file_from_local_to_sandbox` and `download_file_from_sandbox_to_local` support bidirectional file transfer, while `download_file_from_internet_to_sandbox` enables direct retrieval of remote assets from a given URL.

Information Retrieval For knowledge-intensive reasoning, the agent is equipped with two retrieval tools: a Google-based web search tool (`google_search`) that returns structured search results, and a web-scraping tool (`scrape_and_extract_info`) for conditional information extraction from target URLs. Unlike naive webpage scraping, this tool internally leverages a lightweight LLM (*e.g.*, Qwen3-14B [6]) to extract task-relevant information specified by the agent at call time. This mechanism serves as an efficient form of context management, allowing the tool to condense lengthy web or document content into focused textual evidence for subsequent reasoning. Note, to prevent potential information leakage (*e.g.*, searching benchmark answers from HuggingFace), access to HuggingFace has been explicitly disabled in these tools.

3.3. Context Management

To ensure efficient use of the model's context window, we apply two strategies to manage tool responses. These strategies enable our model to perform up to 600 tool calls within a 256K context window.

Recency-Based Context Retention In the standard ReAct paradigm [30], all tool outputs are retained in the message history, often leading to inefficient context utilization. Empirically, we observe that the model's subsequent actions depend primarily on recent observations rather than distant ones. To leverage this recency bias and improve contextual efficiency, we retain only the most recent tool responses while preserving the complete sequence of thoughts and actions. Given a retention budget $K \in \mathbb{N}$ for preserved tool responses, define the index set of the most recent responses at step t as

$$S_t(K) = \{\, i \in \{1, \ldots, t-1\} \mid i \geq t - K \,\}. \tag{4}$$

We construct a recency-filtered history $\widehat{H}_t$ by masking tool responses outside $S_t(K)$:

$$\widehat{H}_t = \big\{ \big(T_i,\ A_i,\ \widehat{O}_i\big) \big\}_{i=1}^{t-1}, \quad \widehat{O}_i \triangleq \begin{cases} O_i, & i \in S_t(K), \\ \varnothing, & \text{otherwise}, \end{cases} \tag{5}$$

where $\varnothing$ denotes that the early tool response is omitted from the context. Subsequent inference is performed on the recency-filtered history:

$$T_t = f_\theta(q, \widehat{H}_t), \qquad A_t = \pi_\theta(\widehat{H}_t, T_t), \tag{6}$$

and upon receiving the new tool response $O_t = \mathrm{Tool}(A_t)$, we update via

$$H_{t+1} = \big\{ (T_1, A_1, O_1), \ldots, (T_t, A_t, O_t) \big\}, \qquad \widehat{H}_{t+1} = \mathrm{Retain}_K(H_{t+1}), \tag{7}$$

where $\mathrm{Retain}_K(\cdot)$ applies the masking rule above with budget K. This recency-based retention strategy preserves the reasoning and action trace while focusing the model's attention on the most contextually relevant observations, thereby freeing additional context for extended reasoning and deeper tool-use trajectories. We found that such a simple context management strategy serves as a strong baseline; it does not lead to degradation in performance and allows the model to have more context space for interactive scaling.

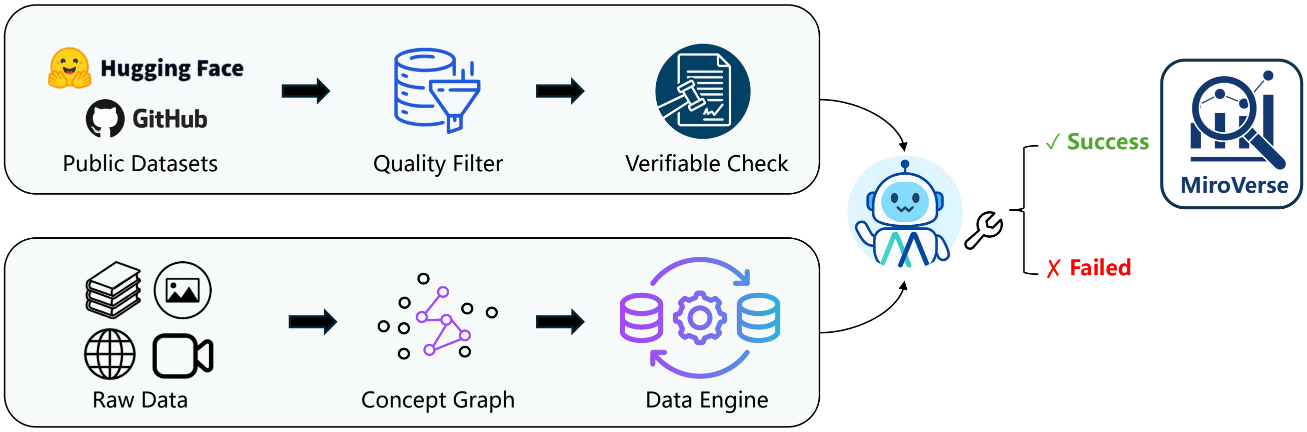


Figure 3: Overview of the data construction pipeline. Public datasets from platforms such as HuggingFace and GitHub are filtered and verified, while raw internet data are processed through knowledge graph generation and a data engine. The resulting QA pairs from both sources are then converted into agentic trajectories, forming the complete MiroVerse v1.0 dataset used for training MiroThinker v1.0.

Result Truncation Certain tools, such as `run_command` and `run_python_code`, may occasionally produce excessively long outputs that could easily overflow the model's context. To mitigate this, we truncate tool responses that exceed a predefined length limit and append the tag "[Result truncated]" at the end to indicate that the content has been shortened.

4. Data Construction

To effectively train our MiroThinker, we construct a large-scale synthetic dataset comprising two main components: (1) multi-document QA (MultiDocQA) synthesis, and (2) agentic trajectory synthesis. These two stages jointly enable the model to acquire both factual grounding and reasoning capabilities.

4.1. MultiDocQA Synthesis

We design a question–answer synthesis pipeline that transforms interlinked web documents into complex, multi-hop QA pairs. The overall process is structured into multiple stages, as described below.

Document Corpus Construction We construct the document corpus from diverse, highly interlinked sources such as Wikipedia, Common Crawl, and curated web repositories, chosen for their rich hyperlink structures and factual reliability. During preprocessing, we clean textual content while preserving hyperlinks, which form the foundation for constructing multi-document reasoning chains. Documents are then categorized into broad knowledge domains via a hybrid method combining metadata extraction and topic modeling, enabling category-aware sampling in later stages.

Document Sampling and Graph Construction We begin by sampling document nodes from our corpus while maintaining balanced representation across different categories. This category-balanced sampling ensures comprehensive coverage across diverse knowledge domains and prevents bias toward over-represented topics. For each sampled seed document, we construct a knowledge graph by following its internal hyperlinks. Specifically, we randomly select one internal link from each document and recursively repeat this process multiple times to build a connected subgraph of related documents.

Document Consolidation After constructing the document graph, we convert each document into markdown format and perform link pruning. We remove all hyperlinks that point to documents outside our selected subgraph, ensuring that the consolidated article maintains coherent references only within the current context. These preprocessed documents are then concatenated into a single comprehensive article that spans multiple related topics while maintaining logical flow through the preserved internal references.

Fact Extraction For each document in the constructed graph, we identify key factual statements that connect back to the central theme established by the seed document. This targeted extraction process ensures that the facts we collect form a coherent knowledge network rather than isolated pieces of information. We prioritize statements that inherently require cross-document reasoning to discover or verify, thereby establishing a foundation for questions that cannot be answered from any single source alone. These extracted facts collectively represent the multi-hop knowledge required to answer complex questions in our dataset.

Constraint Obfuscation To create challenging reasoning scenarios, we systematically obfuscate the extracted facts by transforming them into indirect constraints that require deeper reasoning to resolve. Our obfuscation strategies operate differently depending on the type of information: temporal and spatial details are generalized to broader categories (*e.g.*, March 15, 2023 → in the spring of the 2020s; Paris → a European capital), while other entities and concepts are expressed through referential indirection using related properties or contextual descriptions. This transformation compels models to perform multi-step associative reasoning, integrating knowledge across multiple sources rather than relying on direct fact retrieval.

Question Generation Finally, we prompt a large language model to synthesize questions by selecting and combining multiple obfuscated constraints from our fact pool. The LLM is instructed to generate questions that span diverse domains and require integration of information across different documents in the knowledge graph. This ensures that the resulting questions demand genuine multi-hop reasoning capabilities and cannot be answered through simple pattern matching or single-document retrieval.

4.2. Agentic Trajectory Synthesis

To generate high-quality and diverse agentic trajectory data, we design a multi-layered synthesis framework that integrates multiple agent paradigms, tool invocation mechanisms, and state-of-the-art LLMs.

Agent Paradigms We employ two complementary agent paradigms for trajectory synthesis:

(1) ReAct Single-Agent [30]. This paradigm addresses complex tasks through iterative "think–act–observe" cycles. The agent first analyzes the current state and reasons about the next action, then executes tool calls, and finally updates its internal understanding based on observations. This approach is particularly effective for tasks that require multi-step reasoning and adaptive decision-making.

(2) MiroFlow Multi-Agent [31]. This framework coordinates multiple specialized agents to manage complex workflows. Each agent handles distinct subtasks or domains, communicating through structured protocols. This paradigm produces sophisticated collaborative trajectories that exhibit division of labor, coordination, and emergent collective reasoning.

Tool Invocation Mechanisms To enhance the diversity of synthesized trajectories, we adopt a hybrid approach that combines two complementary tool invocation methods:

(1) Function Calling. A traditional, structured tool invocation approach in which agents interact with external tools through predefined function interfaces. This method provides clear input-output specifications and is suitable for standardized tool usage scenarios.

(2) Model Context Protocol (MCP). A more flexible tool invocation protocol that enables agents to interact with tools more naturally through context negotiation. MCP supports more complex tool composition and dynamic tool discovery, making synthesized trajectories closer to real human-machine interaction patterns.

Diverse Data Synthesis We employ multiple leading LLMs to drive the trajectory synthesis process, including GPT-OSS [32], DeepSeek-V3.1 [5], and other state-of-the-art models. By employing diverse models to generate trajectories, we obtain training data with diverse styles, mitigating single-model biases and ensuring both richness and coverage.

4.3. Open-Source Data Collection

We supplement our synthesized data with diverse open-source QA datasets to broaden coverage and enhance reasoning diversity. The incorporated datasets include MuSiQue [33], HotpotQA [34], WebWalkerQA-Silver [35], MegaScience [36], TaskCraft [37], QA-Expert-Multi-Hop-V1.0 [38], OneGen-TrainDataset-MultiHopQA [39], 2WikiMultihopQA [40], WikiTables [41], WebShaper [14], WebDancer [18], and Toucan-1.5M [42]. We retain only the QA pairs from these datasets and convert them into agentic trajectories through the synthesis pipeline described in Section 4.2. To preserve general conversational abilities, we also include post-training corpora such as AM-Thinking-v1-Distilled [43] and Nemotron-Post-Training-Dataset [44], providing broad coverage of reasoning styles and dialogue forms.

5. Training Pipeline

Based on the open-source Qwen2.5 and Qwen3 models [6], MiroThinker is trained under a three-stage pipeline: (1) Supervised fine-tuning to establish fundamental agentic behaviors; (2) Preference optimization to align decision-making with task objectives; (3) Reinforcement learning to drive creative exploration and generalization in real-world environments.

5.1. Agentic Supervised Fine-tuning

The first stage performs supervised fine-tuning (SFT) to endow MiroThinker with agentic behaviors. The model learns to mimic expert trajectories that involve multi-hop reasoning and tool use.

Data Construction We construct a large-scale SFT dataset $\mathcal{D}_{\text{SFT}} = \{(x_i, H_i)\}_{i=1}^{N}$, where each instance pairs a task instruction x_i with an expert trajectory $H_i = \{(T_{i,t}, A_{i,t}, O_{i,t})\}_{t=1}^{T_i}$ composed of thought–action–observation triplets. During this process, we observe that even when synthesized using leading LLMs, the raw trajectories often contain substantial noise, such as intra-response repetition, cross-response duplication, and invalid tool invocations (*e.g.*, incorrect tool names or malformed arguments). To mitigate these issues, we apply rigorous filtering and data-repair procedures to ensure the consistency and reliability of the final SFT corpus.

Training Objective Each trajectory is treated as a multi-turn dialogue between a *user* and an *assistant*. The user provides the initial task instruction x and subsequently the tool observations O_t, while the assistant produces the reasoning thoughts T_t and tool invocations A_t. During training, tool execution is not actually performed; the observations are pre-recorded and serve as contextual inputs. Given $(x, H) \sim \mathcal{D}_{\text{SFT}}$, the model is trained to predict the expert's thought and action sequences:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,H)}\left[\sum_{t=1}^{T_H} \log \pi_\theta(T_t, A_t \mid x, H_{<t})\right]. \tag{8}$$

This formulation aligns the agent's imitation learning with standard dialogue-style SFT, where tool responses are treated as user turns and the assistant learns to generate the next reasoning or tool call accordingly.

5.2. Agentic Preference Optimization

The second stage refines decision-making through Direct Preference Optimization (DPO) [45], using preference data synthesized from the SFT model.

Data Collection We construct a pairwise preference dataset

$$\mathcal{D}_{\text{PO}} = \{(x_i, H_i^+, H_i^-)\}_{i=1}^{M}, \tag{9}$$

where each task instruction x_i is associated with a preferred trajectory H_i^+ and a dispreferred trajectory H_i^-. Each trajectory represents a full multi-step interaction in thought–action–observation space. Preferences are determined based on the following steps:

(1) Criterion: Correct *vs.* Incorrect Without Enforced Patterns. We construct preference pairs based primarily on the correctness of the final answer. Unlike approaches that rely on handcrafted heuristics or fixed agentic patterns, *e.g.*, predefined planning length, step counts, or reasoning structures, we find that such constraints introduce systematic biases and hinder scalability across diverse tasks and domains. Therefore, we avoid imposing rigid structural formats and instead rely on the correctness of the answer for ranking preferences.

(2) Quality Control: Ensuring Trace Completeness. We further apply strict filtering to ensure the quality and faithfulness of both chosen and rejected trajectories. For a chosen sample, the reasoning trace must be coherent, include an explicit planning process, and yield a clear and correct final answer. For a rejected sample, we similarly require that the trajectory produce a valid final answer. In addition, we apply further filtering to remove surface-level issues such as repetition, truncation, or malformed structures, ensuring that only high-quality trajectories are retained.

Training Objective We refine the SFT model using DPO augmented with an auxiliary SFT loss on preferred trajectories [46, 47], to enhance stability and preserve behavioral consistency. Given a task instruction x and a preference pair (H^+, H^-), the DPO objective encourages the model to assign a higher likelihood to the preferred trajectory while remaining close to the reference SFT model:

$$\mathcal{L}_{\text{DPO}}(x, H^+, H^-) = -\log\sigma\big(\beta\big[(\log\pi_\theta(H^+|x) - \log\pi_\theta(H^-|x)) - (\log\pi_{\text{ref}}(H^+|x) - \log\pi_{\text{ref}}(H^-|x))\big]\big), \tag{10}$$

where π_{ref} is the frozen reference model and β controls deviation from it. The full objective combines DPO with the SFT loss defined above, applied to preferred samples:

$$\mathcal{L}_{\text{PO}}(\theta) = \mathbb{E}_{(x,H^+,H^-)}[L_{\text{DPO}}(x, H^+, H^-)] + \lambda\,\mathcal{L}_{\text{SFT}}^{(+)}(\theta), \tag{11}$$

where $\mathcal{L}_{\text{SFT}}^{(+)}$ denotes the SFT loss on preferred trajectories, and λ controls its weight.

5.3. Agentic Reinforcement Learning

The final stage leverages reinforcement learning to enable the agent to discover creative solutions and adapt to diverse real-world environments through direct interaction and exploration. We employ Group Relative Policy Optimization (GRPO) [48] with fully online policy training by using rollout trajectories to update the policy model exactly once.

Environment Setup We construct a suite of scalable environments capable of supporting thousands of concurrent agentic rollouts, encompassing real-time multi-source search, web scraping and summarization, Python code execution, and Linux VM manipulation. We also built a robust and knowledgeable LLM grading system for verifying noisy agent predictions against ground-truth answers in low latency.

Streaming Rollout Acceleration Unlike single-turn RL tasks like math or reasoning, agentic RL requires a multi-round back-and-force between LLMs and environments, making the completion time of different trajectories heavily long-tailed. The tailness problem is further amplified as our MiroThinkers are capable of interacting with the environments for hundreds of rounds. We implement a rollout mechanism where each agent worker receives prompts from a task queue in a streaming manner until enough completed trajectories are collected for this batch. All unfinished tasks are pushed back to the task queue for the next iteration.

Reward Design Our reward function $R(x, H)$ for a trajectory $H = \{(T_t, A_t, O_t)\}_{t=1}^{T_H}$ given question x combines multiple components:

$$R(x, H) = \alpha_c R_{\text{correct}}(H) - \alpha_f R_{\text{format}}(H), \tag{12}$$

where R_{correct} measures solution correctness and R_{format} penalizes the model for failing to follow format instruction. The coefficients $\{\alpha_c, \alpha_f\}$ are tuned to balance persistent exploration of new solutions with the ability of instruction following.

Trajectory Curation To ensure RL learning signal quality, we implement a comprehensive trajectory filtering pipeline that removes both noisy correct trajectories and trivially incorrect ones. For correct trajectories, we filter out samples exhibiting pathological behaviors such as consecutive API call failures (*e.g.*, more than 5 consecutive network exceptions), redundant retries on identical actions, or excessive environment timeout errors, as these patterns do not reflect genuine problem-solving strategies but rather environmental instabilities. For incorrect trajectories, we discard samples failing due to trivial formatting issues (*e.g.*, missing required answer format) or exhibiting degenerate behaviors like action repetition loops or premature termination without meaningful exploration.

Training Objective GRPO optimizes the policy by sampling multiple trajectories per prompt and computing advantages relative to the group mean. For each prompt x, we sample a group of G trajectories $\{H_1, \ldots, H_G\}, \quad H_i = \{(T_{i,t}, A_{i,t}, O_{i,t})\}_{t=1}^{T_i}$ from the current policy π_θ. The advantage for trajectory H_i is computed as:

$$\hat{A}_i = R(x, H_i) - \frac{1}{G}\sum_{j=1}^{G} R(x, H_j). \tag{13}$$

The GRPO objective maximizes the expected advantage while maintaining proximity to the reference policy:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{H\sim\pi_\theta(\cdot|x)}\left[\hat{A}(x, H) \cdot \log \pi_\theta(H \mid x) - \beta_{\text{KL}} \cdot D_{\text{KL}}\big(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\big)\right], \tag{14}$$

Table 1: Performance comparison across various agent benchmarks. Performance of other Agent Foundation models (AFMs) is collected from Tongyi DeepResearch [11] and model cards of DeepSeek-V3.1, DeepSeek-V3.2, MiniMax-M2, and Kimi-K2. To minimize the impact of randomness from agent-environment interactions on benchmark performance evaluation, we report the average performance and error bars for each benchmark. We use avg@3 for Humanity's Last Exam, BrowseComp, BrowseComp-ZH, WebWalkerQA, and FRAMES, and avg@8 for GAIA, xbench-DeepSearch, and SEAL-0.

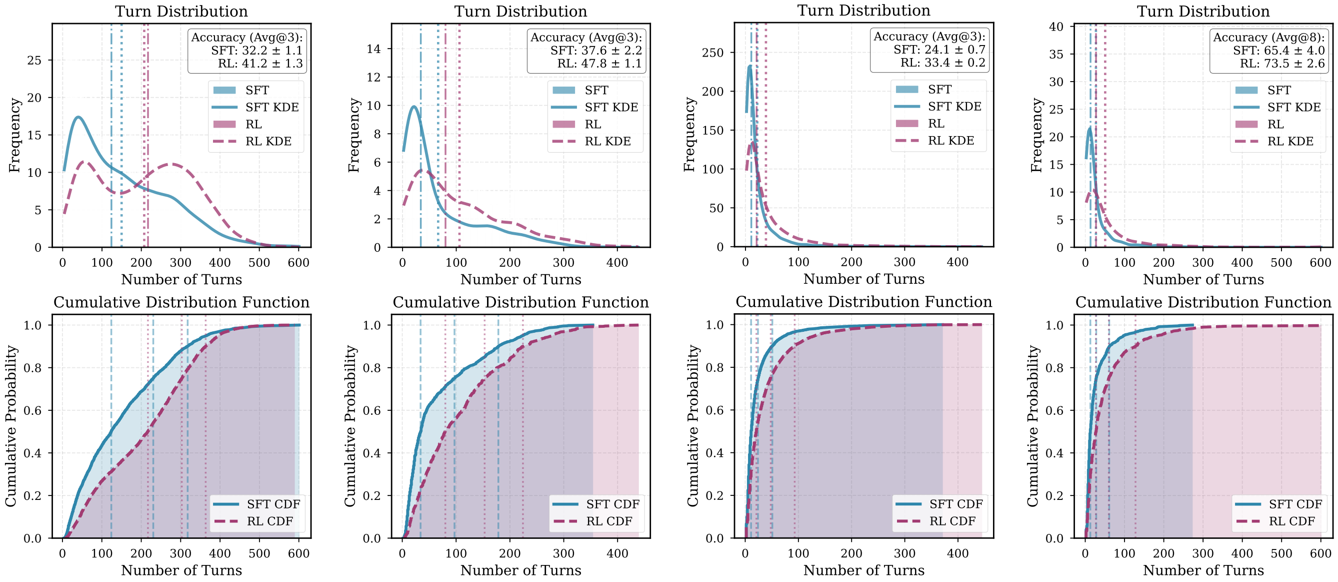
Benchmarks	**Humanity's Last Exam**	**Browse Comp**	**Browse Comp-ZH**	**GAIA**	**xbench DeepSearch**	**WebWalker QA**	**FRAMES**	**SEAL-0**
Foundation Models with Tools								
GLM-4.6 [4]	30.4	45.1	49.5	71.9	70.0	–	–	–
Minimax-M2 [3]	31.8	44.0	48.5	75.7	72.0	–	–	–
DeepSeek-V3.1 [5]	29.8	30.0	49.2	63.1	71.0	61.2	83.7	–
DeepSeek-V3.2 [5]	27.2	40.1	47.9	63.5	71.0	–	80.2	38.5
Kimi-K2-0905 [2]	21.7	7.4	22.2	60.2	61.0	–	58.1	25.2
Claude-4-Sonnet [7]	20.3	12.2	29.1	68.3	64.6	61.7	80.7	–
Claude-4.5-Sonnet [7]	24.5	19.6	40.8	71.2	66.0	–	85.0	53.4
OpenAI-o3 [49]	24.9	49.7	58.1	–	67.0	71.7	84.0	17.1
OpenAI-GPT-5-high [1]	35.2	54.9	65.0	76.4	77.8	–	–	51.4
Research Agents								
OpenAI DeepResearch [24]	26.6	51.5	42.9	67.4	–	–	–	–
ChatGPT-Agent [8]	41.6	68.9	–	–	–	–	–	–
Kimi-Researcher [23]	26.9	–	–	–	69.0	–	78.8	36.0
WebExplorer-8B-RL [21]	17.3	15.7	32.0	50.0	53.7	62.7	75.7	–
DeepMiner-32B-RL [17]	–	33.5	40.1	58.7	62.0	–	–	–
AFM-32B-RL [16]	18.0	11.1	–	55.3	–	63.0	–	–
SFR-DeepResearch-20B [50]	28.7	–	–	66.0	–	–	82.8	–
Tongyi-DeepResearch-30B [11]	32.9	43.4	46.7	70.9	75.0	72.2	90.6	–
MiroThinker-v1.0-8B	21.5±0.4	31.1±1.6	40.2±2.9	66.4±3.2	60.6±3.8	60.6±0.8	80.6±0.5	40.4±2.6
MiroThinker-v1.0-30B	33.4±0.2	41.2±1.3	47.8±1.1	73.5±2.6	70.6±2.2	61.0±0.2	85.4±0.8	46.8±3.2
MiroThinker-v1.0-72B	37.7±0.5	47.1±0.7	55.6±1.1	81.9±1.5	77.8±2.6	62.1±0.6	87.1±0.9	51.0±2.0

where π_{ref} is the reference policy (typically the preference optimization checkpoint), and β_{KL} controls the strength of the KL penalty.

6. Experiments

6.1. Experimental Setup

Evaluation Benchmarks. We use the Qwen2.5 and Qwen3 models [6] as initialization. After training, we evaluate MiroThinker v1.0 models across a diverse suite of agentic benchmarks: Humanity's Last Exam (HLE) [27]; BrowseComp [25] and BrowseComp-ZH [26]; GAIA [28]; xBench-DeepSearch [51]; WebWalkerQA [35]; FRAMES [52]; and SEAL-0 [53]. To ensure fair comparison with prior works, we follow the standard evaluation protocol and report results on the 2,158 text-only subset of Humanity's Last Exam and the 103 text-only subset of GAIA. For other benchmarks, we report the model's performance on the full test set. Note, to prevent potential information leakage (*e.g.*, searching benchmark answers from HuggingFace), access to HuggingFace has been explicitly disabled in these tools.



(a) BrowseComp (b) BrowseComp-ZH (c) HLE (d) GAIA

Figure 4: Illustration of interactive scaling. Reinforcement learning training leads to a substantial increase in the number and depth of agent–environment interactions, resulting in consistently improved task performance across benchmarks. All results are from MiroThinker-v1.0-30B.

Evaluation Protocol. We report all benchmark results using a simple ReAct-style agent to fully demonstrate the strength of our MiroThinkers. We adopt fixed inference settings to guarantee stability and reproducibility: temperature = 1.0, top-p = 0.95, maximum turns = 600, context length = 256K tokens, and maximum output length = 16,384 tokens. For context management, the retention budget is set to 5. For benchmarks exhibiting high per-question variance, we conduct k independent runs and report the averaged score, denoted as avg@k. The specific k values for each benchmark are detailed in the caption of Table 1. All benchmark performances are evaluated using LLM-as-a-Judge. Specifically, GAIA, WebWalkerQA, xBench-DeepSearch, BrowseComp, and BrowseComp-ZH are judged with gpt-4.1-2025-04-14, while Humanity's Last Exam follows its official setup using o3-mini-2025-01-31.

6.2. Overall Performance

MiroThinker establishes a new state-of-the-art on the GAIA benchmark [28] with a score of 81.9%, surpassing the previous leading model, MiniMax-M2 [3], by 6.2 percentage points. The model's capabilities are further demonstrated on the extremely challenging Humanity's Last Exam [27], where it achieves a score of 37.7%. This result outperforms the proprietary state-of-the-art model, GPT-5-high [1], by 2.5 percentage points while utilizing the identical Python and search toolset.

Furthermore, MiroThinker delivers highly competitive performance on the BrowseComp [25] and SEAL-0 [53] benchmarks, with scores of 47.1% and 51.0% respectively, placing it on par with advanced proprietary systems such as OpenAI DeepResearch [24], OpenAI o3 [49], and Anthropic Claude 4.5 [7]. In the domain of multilingual research, MiroThinker shows exceptional mastery by setting new open-source records on Chinese benchmarks, achieving 55.6% on BrowseComp-ZH [26] and 77.8% on xbench-DeepSearch [51].

Our commitment to excellence extends across all scales. The 8B and 30B variants of MiroThinker also achieve state-of-the-art performance within their respective size classes, providing the community with access to powerful deep research models at various model scales.

6.3. Interactive Scaling

We examine how RL reshapes agent–environment interaction patterns. As shown in Figure 4, the RL-tuned MiroThinker-v1.0-30B model exhibits substantially longer and deeper interaction trajectories than its SFT counterpart across BrowseComp [25], BrowseComp-ZH [26], HLE [27], and GAIA [28]. Guided by verifiable rewards, RL enables the model to explore more exhaustive solution paths with significantly greater interaction depth, systematically probing multiple strategies and validating intermediate results before reaching conclusions.

This behavioral shift directly correlates with improved accuracy (Avg@3,8), yielding 8–10 point gains on average. We refer to this consistent relationship between interaction depth and performance as the interactive scaling: as the frequency and depth of tool-augmented interactions increase, research reasoning capability improves correspondingly. This forms the third dimension of scaling, alongside model size and context length, defining MiroThinker's pathway toward more general agentic intelligence.

6.4. Limitations

We have identified several limitations in the current version of the model, which we plan to address in future updates. These limitations include the following:

Tool-Use Quality under Interactive Scaling Interactive scaling enables richer and more complex tool interactions, but also exposes limitations in tool-use quality. We find that the RL-tuned model invokes external tools more frequently than the SFT model, yet a portion of these invocations yield marginal or redundant contributions. This indicates that scaling improves agentic performance, but further optimization is needed to enhance tool-use efficiency and action quality.

Overlong Chain-of-Thought Reinforcement learning tends to encourage the model to produce longer responses to improve accuracy, which can result in excessively long, repetitive, and less readable reasoning chains. This, in turn, slows down task completion and degrades user experience.

Language Mixing For non-English inputs, the model's responses may exhibit multilingual mixing. For instance, when a user query is in Chinese, the model's internal reasoning or intermediate outputs may contain a blend of English and Chinese elements, which may lead to suboptimal performance in Chinese.

Limited Sandbox Capability The model is not yet fully proficient in utilizing code-execution and file-management tools. It may occasionally generate code or commands that lead to sandbox timeouts, or misuse the code-execution tool to read web pages or PDFs, tasks that would be far more efficiently handled by dedicated web-scraping tools. Moreover, the model sometimes demonstrates insufficient familiarity with sandbox ID management, frequently forgetting to initialize a sandbox before invoking related operations.

7. Conclusions

We introduce MiroThinker v1.0, an open-source research agent that advances tool-augmented reasoning through model, context, and interactive scaling. By extending scaling into the interaction dimension, MiroThinker shows that research capability improves not only with larger models or longer context, but also with deeper and more frequent agent–environment interactions that enable error correction and knowledge acquisition. Our experiments demonstrate predictable gains from interactive scaling across diverse benchmarks, establishing interaction depth as a third critical axis for building next-generation research agents. We hope MiroThinker provides a strong baseline and an open platform for further exploration of interaction-scaled, agentic intelligence.

References

[1] OpenAI. Introducing gpt-5. https://openai.com/index/introducing-gpt-5/, 2025.

[2] Kimi, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

[3] MiniMax AI. Minimax m2 & agent: Ingenious in simplicity. https://www.minimax.io/news/minimax-m2, 2025.

[4] Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

[5] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[6] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[7] Anthropic. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5, 2025.

[8] OpenAI. Introducing chatgpt agent: bridging research and action. https://openai.com/index/introducing-chatgpt-agent/, 2025.

[9] Anthropic. Claude takes research to new places. https://claude.com/blog/research, 2025.

[10] Meituan LongCat Team, Bei Li, Bingye Lei, Bo Wang, Bolin Rong, Chao Wang, Chao Zhang, Chen Gao, Chen Zhang, Cheng Sun, et al. Longcat-flash technical report. *arXiv preprint arXiv:2509.01322*, 2025.

[11] Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, et al. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025.

[12] Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*, 2025.

[13] Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025.

[14] Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, et al. Webshaper: Agentically data synthesizing via information-seeking formalization. *arXiv preprint arXiv:2507.15061*, 2025.

[15] Tianqing Fang, Zhisong Zhang, Xiaoyang Wang, Rui Wang, Can Qin, Yuxuan Wan, Jun-Yu Ma, Ce Zhang, Jiaqi Chen, Xiyun Li, et al. Cognitive kernel-pro: A framework for deep research agents and agent foundation models training. *arXiv preprint arXiv:2508.00414*, 2025.

[16] Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang, Qianben Chen, Weichen Sun, Qiexiang Wang, et al. Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic rl. *arXiv preprint arXiv:2508.13167*, 2025.

[17] Qiaoyu Tang, Hao Xiang, Le Yu, Bowen Yu, Yaojie Lu, Xianpei Han, Le Sun, WenJuan Zhang, Pengbo Wang, Shixuan Liu, et al. Beyond turn limits: Training deep search agents with dynamic context window. *arXiv preprint arXiv:2510.08276*, 2025.

[18] Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, et al. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*, 2025.

[19] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.

[20] Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

[21] Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, et al. Webexplorer: Explore and evolve for training long-horizon web agents. *arXiv preprint arXiv:2509.06501*, 2025.

[22] Gongrui Zhang, Jialiang Zhu, Ruiqi Yang, Kai Qiu, Miaosen Zhang, Zhirong Wu, Qi Dai, Bei Liu, Chong Luo, Zhengyuan Yang, et al. Infoagent: Advancing autonomous information-seeking agents. *arXiv preprint arXiv:2509.25189*, 2025.

[23] Moonshot AI. Kimi-researcher: End-to-end rl training for emerging agentic capabilities. https://moonshotai.github.io/Kimi-Researcher/, 2025.

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

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

[26] Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, et al. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*, 2025.

[27] Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

[28] Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

[29] xAI. Grok 3 beta — the age of reasoning agents. https://x.ai/news/grok-3, 2025.

[30] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2022.

[31] MiroMind AI Team. Miroflow: A high-performance open-source research agent framework. https://github.com/MiroMindAI/MiroFlow, 2025.

[32] Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

[33] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[34] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, 2018.

[35] Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, et al. Webwalker: Benchmarking llms in web traversal. *arXiv preprint arXiv:2501.07572*, 2025.

[36] Run-Ze Fan, Zengzhi Wang, and Pengfei Liu. Megascience: Pushing the frontiers of post-training datasets for science reasoning. *arXiv preprint arXiv:2507.16812*, 2025.

[37] Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Liu, et al. Taskcraft: Automated generation of agentic tasks. *arXiv preprint arXiv:2506.10055*, 2025.

[38] Khai Mai. Qa-expert-multi-hop-qa-v1.0. https://huggingface.co/datasets/khaimaitien/qa-expert-multi-hop-qa-V1.0, 2023.

[39] ZJUNLP. Onegen-traindataset-multihopqa. https://huggingface.co/datasets/zjunlp/OneGen-TrainDataset-MultiHopQA, 2024.

[40] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

[41] Sunjun Kweon, Yeonsu Kwon, Seonhee Cho, Yohan Jo, and Edward Choi. Open-wikitable: Dataset for open domain question answering with complex reasoning over table. *arXiv preprint arXiv:2305.07288*, 2023.

[42] Zhangchen Xu, Adriana Meza Soria, Shawn Tan, Anurag Roy, Ashish Sunil Agrawal, Radha Poovendran, and Rameswar Panda. Toucan: Synthesizing 1.5 m tool-agentic data from real-world mcp environments. *arXiv preprint arXiv:2510.01179*, 2025.

[43] Xiaoyu Tian, Yunjie Ji, Haotian Wang, Shuaiting Chen, Sitong Zhao, Yiping Peng, Han Zhao, and Xiangang Li. Not all correct answers are equal: Why your distillation source matters. *arXiv preprint arXiv:2505.14464*, 2025.

[44] Aarti Basant, Abhijit Khairnar, Abhijit Paithankar, Abhinav Khattar, Adithya Renduchintala, Aditya Malte, Akhiad Bercovich, Akshay Hazare, Alejandra Rico, Aleksander Ficek, et al. Nvidia nemotron nano 2: An accurate and efficient hybrid mamba-transformer reasoning model. *arXiv preprint arXiv:2508.14444*, 2025.

[45] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[46] Zhihan Liu, Miao Lu, Shenao Zhang, Boyi Liu, Hongyi Guo, Yingxiang Yang, Jose Blanchet, and Zhaoran Wang. Provably mitigating overoptimization in rlhf: Your sft loss is implicitly an adversarial regularizer. *Advances in Neural Information Processing Systems*, 37:138663–138697, 2024.

[47] Weiyun Wang, Zhe Chen, Wenhai Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Jinguo Zhu, Xizhou Zhu, Lewei Lu, Yu Qiao, et al. Enhancing the reasoning ability of multimodal large language models via mixed preference optimization. *arXiv preprint arXiv:2411.10442*, 2024.

[48] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[49] OpenAI. Introducing openai o3 and o4-mini. https://openai.com/zh-Hans-CN/index/introducing-o3-and-o4-mini/, 2025.

[50] Xuan-Phi Nguyen, Shrey Pandit, Revanth Gangi Reddy, Austin Xu, Silvio Savarese, Caiming Xiong, and Shafiq Joty. Sfr-deepresearch: Towards effective reinforcement learning for autonomously reasoning single agents. *arXiv preprint arXiv:2509.06283*, 2025.

[51] Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, et al. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations. *arXiv preprint arXiv:2506.13651*, 2025.

[52] Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4745–4759, 2025.

[53] Thinh Pham, Nguyen Nguyen, Pratibha Zunjare, Weiyuan Chen, Yu-Min Tseng, and Tu Vu. Sealqa: Raising the bar for reasoning in search-augmented language models. *arXiv preprint arXiv:2506.01062*, 2025.