

# Axion 1.0: Open Agent Foundation Models, Simplified

Yuqi Jia<sup>1</sup>, Yunjie Tian<sup>2,†</sup>, Jiayu Zheng<sup>2</sup>  
Yijun Liang<sup>3</sup>, Jingchen Sun<sup>4</sup>, Mengyu Zhao<sup>2</sup>, Di Fu<sup>2</sup>

<sup>1</sup>Duke University, <sup>2</sup>ByteDance, <sup>3</sup>University of Maryland, College Park, <sup>4</sup>University at Buffalo

<sup>†</sup>Project Lead

## Abstract

We introduce **Axion 1.0**, a fully open agent foundation model that combines strong and broad agentic capabilities with a remarkably simple training recipe. Axion 1.0 is trained in only three stages: two lightweight supervised fine-tuning stages followed by reinforcement learning, without dedicated mid-training, direct preference optimization (DPO), or on-policy distillation (OPD). The first SFT stage develops strong self-contained reasoning through model-aware data construction, where we synthesize challenging problems and adversarially strengthen them based on the base model’s own successful solution strategies. The second stage extends these capabilities to tool-dependent reasoning through a scalable pipeline for constructing diverse and automatically verifiable tasks grounded in real-world data sources and deterministic computation. Finally, reinforcement learning further strengthens long-horizon reasoning and interaction with external tools and environments. Despite this compact training recipe, Axion 1.0 achieves strong performance across a broad suite of agentic benchmarks and is competitive with leading commercial models on several of them. Equally importantly, Axion 1.0 is open beyond model weights: we release the complete recipe, including data construction, generation, filtering, verification, prompts, and training code, enabling the entire development process to be reproduced and extended. Axion 1.0 demonstrates that strong agent foundation models can be both *open* and *simple*, offering a practical and reproducible path toward advancing open agentic intelligence.

**Project:** <https://axion1-0.github.io/>

**Code:** [https://github.com/bytedance/axion1\\_0](https://github.com/bytedance/axion1_0)

## 1 Introduction

Large language models are rapidly evolving beyond response generation into agents capable of reasoning, acting, and interacting with external tools and environments [35, 48]. This transition is already evident in widely deployed commercial AI systems, such as ChatGPT [25, 29] and Claude [2, 6], which increasingly operate as agentic systems rather than relying solely on the outputs of standalone language models. By integrating capabilities such as web search, code execution, and tool invocation, these systems can tackle complex tasks through extended, multi-step trajectories: decomposing objectives, acquiring information, taking actions, interpreting observations, and iteratively refining their solutions. This evolution is also driving the development of *agent foundation models*, where reasoning, tool use, and sustained interaction become intrinsic model capabilities rather than functionality provided solely by external scaffolding. Recent open and proprietary models have demonstrated remarkable progress across reasoning, search, coding, and other

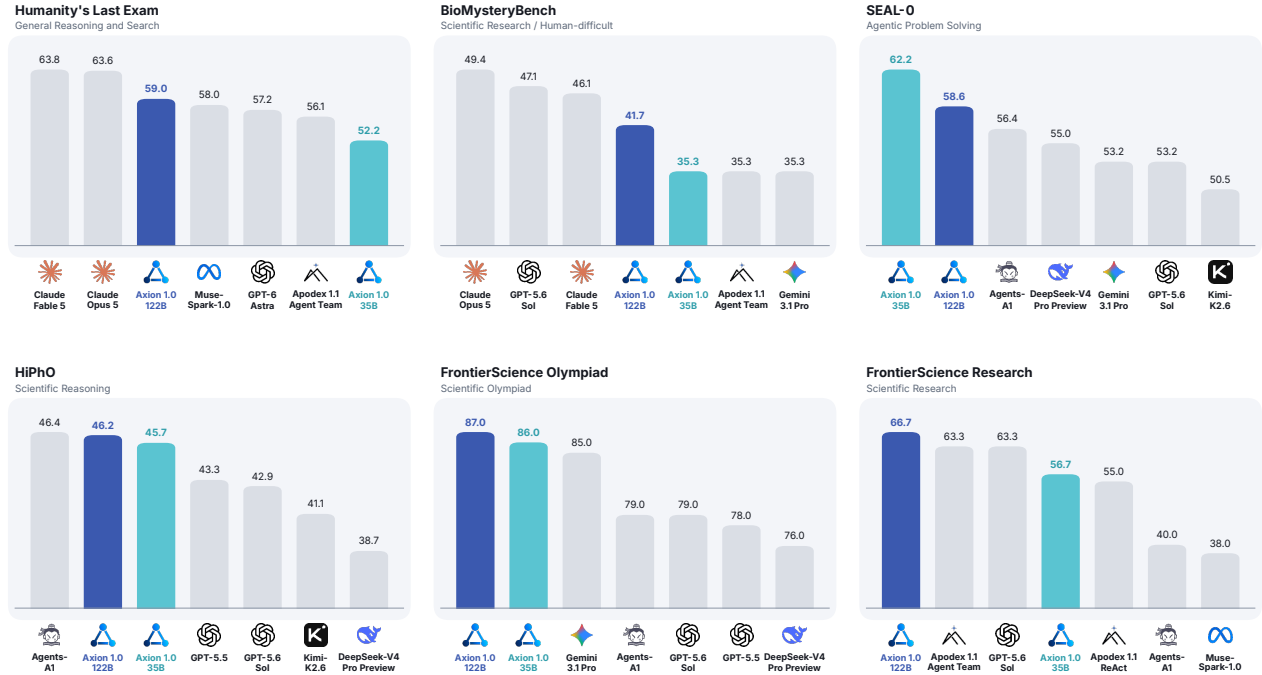


**Figure 1 Performance comparison of Axiom 1.0 with leading open and proprietary models** across six challenging benchmarks, spanning general reasoning, scientific reasoning, search, and agentic problem solving. Blue and Teal bars represent Axiom 1.0-122B and Axiom 1.0-35B, respectively, while gray bars denote competing models. Despite its simple three-stage training recipe, Axiom 1.0 achieves competitive or superior performance against frontier models across diverse reasoning and agentic tasks.

tool-augmented tasks [1, 6, 7, 10, 12, 22, 23, 29, 45].

At the same time, the training pipelines behind these capabilities are becoming increasingly elaborate. Modern agent models often rely on continued or mid-training, multiple supervised stages, specialized capability-specific data mixtures, preference optimization, reinforcement learning, and additional post-training procedures [1, 7, 22, 41, 45]. While individual components may be well motivated, their accumulation makes it increasingly difficult to understand which ingredients are truly necessary. Such complexity also raises the cost of reproduction, obscures the sources of capability gains, and makes training recipes harder to transfer across models. Moreover, many “open” models release their weights while leaving substantial parts of the development process—including data construction, filtering, verification, prompts, and training procedures—unavailable.

In this work, we ask a simple question: *how much of this complexity is actually necessary to build a strong and broadly capable agent foundation model?* We introduce **Axiom 1.0**, a fully open agent foundation model developed with a deliberately simple training recipe. Axiom 1.0 is trained in only three stages: two lightweight supervised fine-tuning stages followed by reinforcement learning, without dedicated mid-training, direct preference optimization (DPO), or on-policy distillation (OPD). The first SFT stage develops strong self-contained reasoning, the second extends these capabilities to reasoning that requires interaction with external tools, and reinforcement learning further strengthens long-horizon reasoning and agent–environment interaction.

A central principle behind Axiom 1.0 is that simplifying the training pipeline does not require simplifying the learning problem. Instead, we place greater emphasis on the construction of high-quality and challenging training data. For self-contained reasoning, we develop a model-aware data construction process that synthesizes difficult problems and further strengthens them according to the base model’s existing capabilities. For tool-dependent reasoning, we construct diverse tasks grounded in real-world data sources and deterministic computation, enabling scalable training with automatically verifiable outcomes. Together, these two stages establish a progression from independent reasoning to grounded interaction, while reinforcement learning further consolidates these capabilities over longer trajectories. We describe the corresponding data construction

and training procedures in detail in later sections.

Despite its compact training recipe, Axion 1.0 develops broad and strong agentic capabilities. As summarized in Figure 1, we evaluate Axion 1.0 across a diverse suite of benchmarks covering reasoning, tool use, information seeking, coding, and long-horizon agentic interaction. Axion 1.0 achieves competitive performance against leading open models and several frontier commercial models across these evaluations. These results suggest that strong agentic capability need not depend on an increasingly fragmented sequence of specialized training stages.

Axion 1.0 is also designed to be open beyond model weights. We release the complete development recipe, including data construction, generation, filtering, verification, prompts, and training code. Our goal is not only to make the final model accessible, but to make its development process reproducible, inspectable, and extensible. This distinction is particularly important for agent foundation models, whose capabilities often depend as much on training data and interaction design as on the model architecture itself.

Our main contributions are summarized as follows:

- We introduce **Axion 1.0**, a fully open agent foundation model trained with a simple three-stage pipeline consisting of two lightweight SFT stages followed by reinforcement learning, without dedicated mid-training, DPO, or OPD stages.
- We develop effective data construction pipelines for both **self-contained reasoning** and **tool-dependent reasoning**, combining model-aware task strengthening with scalable construction of diverse and automatically verifiable agentic tasks.
- We show that this compact recipe produces **strong and broad agentic capabilities**, with Axion 1.0 achieving competitive performance against leading open models and several frontier commercial models across a broad benchmark suite.
- We release the **complete development recipe**—including data construction, generation, filtering, verification, prompts, and training code—to support reproducible and extensible research on agent foundation models.

Overall, Axion 1.0 demonstrates that strong agent foundation models can be both *simple to build* and *open to reproduce*. Rather than continually expanding the training stack, our results suggest that carefully constructed data, a clear progression from reasoning to interaction, and reinforcement learning over agentic trajectories can provide a practical path toward capable and reproducible open agents.

## 2 Data Construction

We construct two complementary SFT datasets: one for *self-contained reasoning* and the other for *tool-dependent reasoning*. The first teaches the model to solve challenging problems using the information provided in the question, whereas the second teaches it to acquire missing information through external tools and reason over the retrieved evidence. Both pipelines separate *problem construction* from *trajectory collection*: we first establish a question and a verifiable reference answer, then retain only correctly solved trajectories for training. Figure 2 summarizes the process.

### 2.1 Self-Contained Reasoning Tasks

**Seed problems and initial difficulty enhancement:** We collect question–answer pairs from public course materials, competitions, and reasoning datasets spanning mathematics, physics, chemistry, biology, and computer science. Rather than training directly on these potentially familiar problems, we use them as seeds for generating new, more challenging questions. For each seed, a generator LLM first applies *composition* (operation A): it identifies a quantity established by the seed’s solution and constructs a new, self-contained question that requires this quantity as an essential intermediate result. The new question must have an unambiguous, checkable final answer and must not reveal the original seed or its answer. This produces the initial pool of enhanced problems.

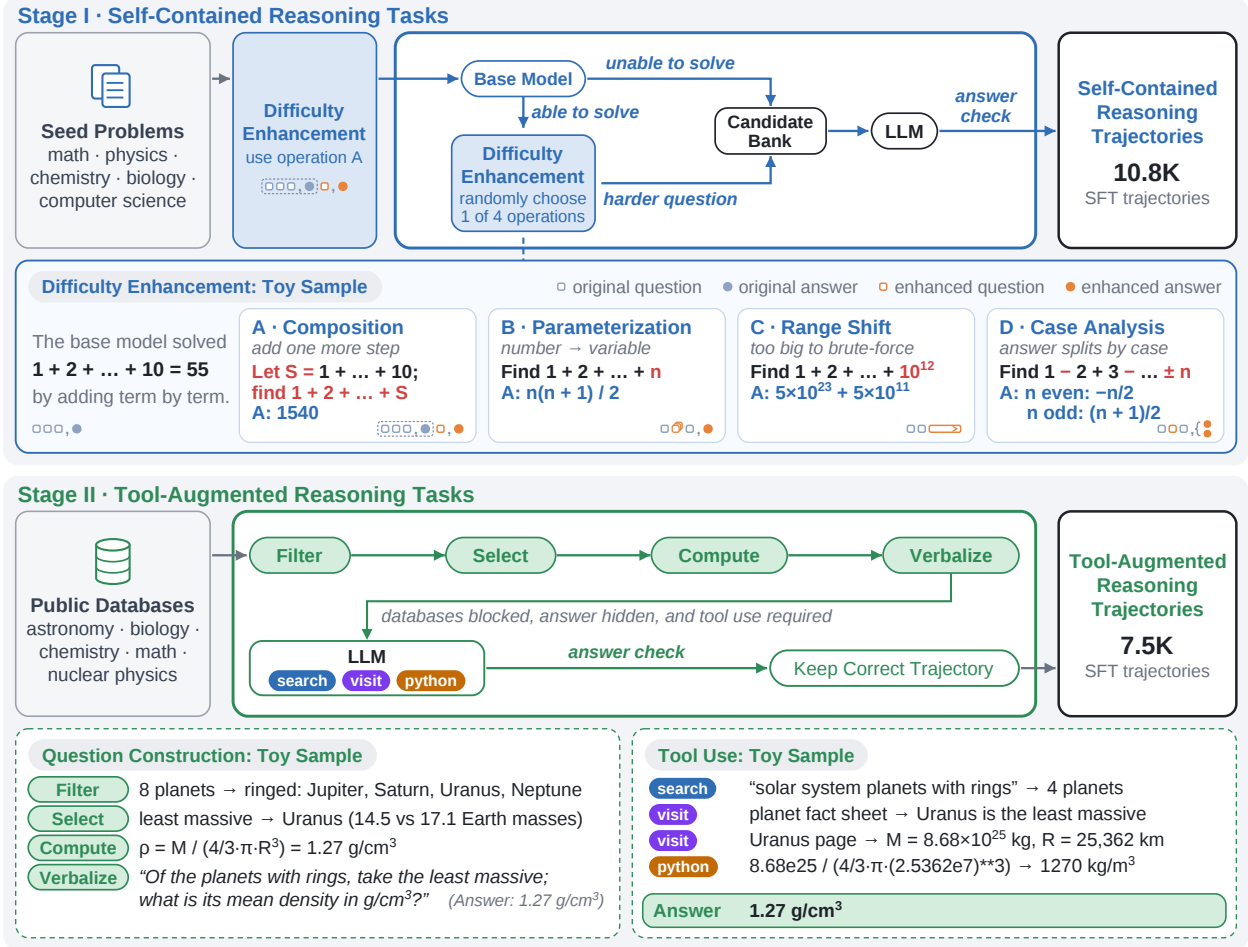


**Figure 2 Overview of the two-stage data construction pipeline.** **Stage I** constructs self-contained scientific reasoning problems from public seeds. An initial composition step increases their difficulty; problems solved by the base model are further strengthened using one of four transformations (A–D). An LLM then generates solutions, from which correct reasoning trajectories are retained. **Stage II** constructs tool-dependent questions through deterministic filtering, selection, and computation over structured data sources. The construction-time answers and direct database access are withheld during solving, and only verified tool-use trajectories are retained for SFT.

**Base-model feedback and targeted enhancement:** We evaluate each validated enhanced problem with the base model, allowing Python execution but disabling search and webpage visits. A problem the base model answers incorrectly is added directly to the candidate question bank. If the base model answers correctly, we provide its successful solution to the generator LLM and randomly apply *one* of four difficulty-enhancement operations:

- **A. Composition:** Add a further dependent reasoning step whose solution requires the answer to the current problem as an intermediate result, forming a new question with a new final answer.
- **B. Parameterization:** Replace a fixed value with a symbolic parameter, requiring a general expression rather than a single numerical result.
- **C. Range Shift:** Enlarge or shift the range of a parameter so that simple enumeration or a shortcut used in the original solution is no longer sufficient.
- **D. Case Analysis:** Introduce conditions under which the correct answer differs across cases, requiring the model to identify and resolve each case.

The transformed question is added to the candidate bank after answer verification; we do not require the base model to fail a second test before admitting it. In particular, composition can appear twice: once to create the initial problem from the seed, and again as a possible enhancement when the base model solves that problem. The objective is to increase the reasoning required, not merely the length or numerical precision of a question.

**Reference-answer verification:** Every newly generated or rewritten problem is checked independently of the base model’s prediction. GPT-5.5 [27] independently solves the problem three times. We accept the reference answer only when all three solutions agree and their derivations pass correctness checks; ill-posed or underspecified problems are discarded. Importantly, an answer is re-established after each rewrite rather than inherited from the previous question. The resulting bank therefore combines initially enhanced problems the base model could not solve with further-strengthened versions of problems it could solve.

**Trajectory collection and filtering:** A separate LLM solves the candidate questions with Python available and external retrieval disabled. Each trajectory contains the model’s reasoning, any Python calls and their outputs, and the final answer. We compare the final answer against the verified reference and retain only correct, complete trajectories with valid tool interactions and output format; duplicate questions are removed. The resulting self-contained reasoning dataset is

$$\mathcal{D}_{\text{sc}} = \{(x_i, \tau_i, \emptyset, y_i)\}_{i=1}^{M_{\text{sc}}}, \quad (1)$$

where  $x_i$  is the question,  $y_i$  is its reference answer,  $\tau_i$  is the retained solution trajectory, and  $\emptyset$  indicates that no separate programmatic intermediate verification targets are attached.

## 2.2 Tool-Dependent Reasoning Tasks

Unlike Stage I, Stage II primarily targets problems whose answers depend on facts that are not provided in the question. We assemble structured, publicly available data sources covering domains such as biology, chemistry, mathematics, geography, and aviation. Each source supplies records with identifiable entities, attributes, and associated values. We use deterministic programs to construct questions and compute their reference answers, ensuring that the tasks are verifiable before an LLM attempts them. A smaller set of mathematical and algorithmic tasks is generated through deterministic computation and checked by independent implementations.

**Filter-select-compute construction:** For database-grounded tasks, each question is built through four steps:

1. **Filter:** Apply explicit conditions to the source records to produce a candidate set of entities.
2. **Select:** Identify exactly one entity by a criterion such as the largest or smallest value of an attribute within that set.
3. **Compute:** Retrieve additional attributes of the selected entity and calculate a derived quantity using a deterministic formula, including any necessary unit conversions.
4. **Verbalize:** Use an LLM to turn the selection conditions and requested computation into one natural-language question, while preserving the programmatically determined reference answer.

For example, we can filter for planets with rings, select the planet with the smallest mass, and ask for its mean density. The solver must first identify Uranus, then find its mass and radius, and finally compute  $\rho = M/(4\pi R^3/3)$ , yielding approximately 1.27 g/cm<sup>3</sup>. This illustrates how a single question combines entity identification, factual retrieval, and computation rather than asking for a directly memorized fact.

**Deterministic validation and question formulation:** At construction time, the program checks that each filter produces valid records, the selection criterion identifies a unique entity, and the required attributes are available. When selecting an extreme value, we also require a sufficient margin over the next candidate to avoid ambiguous or unstable answers. For tasks spanning multiple databases, entity identifiers and cross-references connect successive retrieval steps; the output of one step determines the next. Deterministic calculations establish the final answer, with independent implementations used for computational tasks when applicable.

The verbalization LLM may change phrasing but must preserve all selection constraints, values, units, and requested precision. We check the rewritten question for dropped constraints and accidental disclosure of the selected entity or answer. When possible, we also avoid naming the source database, so the question does not reveal a direct lookup path.

**Enforcing tool dependence:** The databases used to construct the questions and their stored answers are available to the construction pipeline but are not exposed to the model during trajectory collection. In particular, we block direct access to the construction-time database tables while allowing general-purpose search, webpage visits, and Python. The LLM must therefore recover the necessary facts from accessible external information and combine them to reach the answer. We additionally probe database-grounded questions without retrieval tools and remove those that can already be answered correctly in a closed-book setting. These measures focus the supervision on information seeking and evidence-based computation rather than recall of the reference answer.

**Trajectory collection and filtering:** An LLM solves each retained question using search, webpage visits, and Python as needed. We compare its final answer with the reference answer computed during construction, using exact matching or task-appropriate numerical tolerances. Only correct trajectories with valid tool calls, complete termination, and consistent formatting are included in SFT. We also preserve the intermediate entities and values established by the construction program as optional verification targets, without revealing them during solving. The resulting dataset is

$$\mathcal{D}_{\text{ta}} = \{(x_i, \tau_i, z_i, y_i)\}_{i=1}^{M_{\text{ta}}}, \quad (2)$$

where  $z_i = \{v_{i,1}, \dots, v_{i,m_i}\}$  contains the intermediate verification targets associated with the question. Together, the two datasets support a progression from independent scientific reasoning to reasoning grounded in retrieved information and tool use.

## 3 Training

### 3.1 Overview

Axion 1.0 follows a three-stage training recipe: two consecutive stages of supervised fine-tuning (SFT), followed by reinforcement learning (RL). The first SFT stage develops self-contained reasoning, the second introduces tool-dependent problem solving, and RL further improves decision-making over the model’s own tool-interaction trajectories. We use the data constructed in Section 2 without additional mid-training, direct preference optimization (DPO), or on-policy distillation (OPD). The infrastructure supporting long-trajectory SFT and asynchronous RL is described in Section 5.

### 3.2 Two-Stage Supervised Fine-Tuning

We train the model to reason before requiring it to retrieve and integrate external information. In **Stage I**, we fine-tune on the self-contained reasoning dataset  $\mathcal{D}_{\text{sc}}$  (Section 2.1), where solutions may use Python but not search or webpage visits. In **Stage II**, we continue training from the Stage I checkpoint on the tool-dependent dataset  $\mathcal{D}_{\text{ta}}$  (Section 2.2). With construction-time databases withheld, these trajectories demonstrate how to search for missing facts, inspect retrieved pages, and combine external evidence with reasoning and computation. We analyze the contribution of each training stage in Section 4.3.

**Assistant-only supervision:** For both datasets, each training example contains a question  $x$ , a trajectory  $\tau$ , a reference answer  $y$ , and verification targets  $z$  (empty for Stage I). We serialize the complete interaction, including tool observations, but supervise only assistant-generated tokens: reasoning, tool calls, and final answers. System and user messages and tool observations provide context without contributing to the loss. The reference answer and verification targets are used for checking trajectories and are not included in the

model input. The SFT objective is

$$\mathcal{L}_{\text{SFT}}(\theta; \mathcal{D}_k) = - \sum_{(x, \tau, z, y) \in \mathcal{D}_k} \sum_{j \in \mathcal{I}_{\text{asst}}(\tau)} \log \pi_{\theta}(\tau_j \mid x, \tau_{<j}), \quad (3)$$

for  $k \in \{\text{sc}, \text{ta}\}$ , where  $\pi_{\theta}$  denotes the model’s conditional token distribution and  $\mathcal{I}_{\text{asst}}(\tau)$  indexes assistant-token positions in the interaction history.

### 3.3 Reinforcement Learning

SFT teaches the model from successful demonstrations, but does not directly optimize its decisions when generating and acting on its own. Starting from the Stage II checkpoint, we therefore apply group-relative policy optimization (GRPO) [9] with the `verl` framework [37]. The model generates trajectories while interacting with search, webpage visits, scholarly retrieval, and Python tools, with context management supporting long-horizon rollouts. RL tasks come from the programmatic construction pipeline in Section 2.2, which provides both verifiable final answers and intermediate targets. This allows us to reward correct outcomes while giving partial credit for meaningful progress on unfinished solutions.

**Outcome and process rewards:** For a task with question  $x$ , reference answer  $y$ , and verification targets  $z$ , let  $\tau$  be a sampled trajectory. We define  $F(\tau; y) \in \{0, 1\}$  to indicate whether the final answer is correct,  $A(\tau) \in \{0, 1\}$  to indicate whether a valid final answer is submitted, and  $P(\tau; z) \in [0, 1]$  to measure verified intermediate progress. Writing  $F$ ,  $A$ , and  $P$  for these trajectory-specific values, we define the reward as

$$R(\tau; y, z) = \begin{cases} 1, & F = 1, \\ \max(0, 0.3P - 0.1(1 - A)), & F = 0. \end{cases} \quad (4)$$

A correct final answer receives the full reward. Otherwise, validated intermediate progress can receive partial credit, while the non-submission penalty discourages trajectories that make progress without producing a final answer.

**Verification of intermediate progress:** For an ordered chain of  $m \geq 1$  dependent targets  $v_1, \dots, v_m$  recorded in  $z$ , let  $b_j(\tau; z) \in \{0, 1\}$  indicate whether the trajectory correctly establishes target  $v_j$ . We aggregate these checks in dependency order:

$$P(\tau; z) = \frac{1}{m} \sum_{j=1}^m \prod_{\ell=1}^j b_{\ell}(\tau; z), \quad (5)$$

so a later step earns credit only when all its prerequisites have been verified. For example, if only the first and third checks pass in a three-step chain, then  $P(\tau; z) = 1/3$ . Tasks without intermediate targets receive  $P(\tau; z) = 0$ .

We require intermediate results to be identified or used in the agent’s reasoning, rather than merely appearing among retrieved candidates. Targets that require external grounding, such as entity identifiers, must also be supported by tool observations. Verification combines deterministic checks with language-model judgments of semantic equivalence. Numerical tolerances are computed programmatically and passed explicitly to the judge, while rule-based matching extracts relevant trajectory passages for evaluating intermediate steps.

**RL task selection:** We select RL questions from the same programmatically constructed task pool, retaining the question  $x$ , reference answer  $y$ , and intermediate targets  $z$ . We exclude questions that overlap with held-out evaluations or have known answer or formatting defects. For each remaining question, we sample four trajectories from the current model and score them using Equation 4. We retain the question only if at least one trajectory reaches the correct final answer and the four rewards have nonzero variance. This favors tasks with a demonstrated solution path and a useful group-relative learning signal.

## 4 Evaluation

We evaluate Axion 1.0 along two dimensions: its performance relative to existing open-weight and proprietary models, and the capability gains associated with our three-stage training recipe (Section 3). We first present the evaluation protocol and overall results, then analyze successive training checkpoints, base-model choice, and performance across model sizes.

### 4.1 Evaluation Setup

**Models:** We evaluate two main Axion 1.0 configurations: 35B-A3B and 122B-A10B, initialized from Qwen3.6-35B-A3B and Qwen3.5-122B-A10B, respectively. We also evaluate their base models and intermediate training checkpoints to examine the effects of successive stages. To assess transfer to smaller models, we train 4B and 9B variants with the two SFT stages, without RL. Our models are evaluated in a ReAct environment supporting web search, webpage visits, scholarly retrieval, and Python execution, subject to benchmark-specific restrictions.

**Training settings:** We use full-parameter SFT with sample packing. RL uses asynchronous GRPO with eight rollouts per question and session-level training [45], where context-condensation events divide each rollout into sessions sharing its reward and group-relative advantage. The training infrastructure and systems-level optimizations behind these runs are described in Section 5, and detailed hyperparameters are provided in Tables 4 and 5 of Appendix A.

**Benchmarks:** We report results on six evaluation settings: Humanity’s Last Exam (HLE) [8], BioMystery-Bench (BMB) [3], SEAL-0 [30], HiPhO [49], and the Olympiad and Research splits of FrontierScience [26]. HLE measures expert-level problem solving across disciplines; SEAL-0 emphasizes information seeking; and HiPhO and FrontierScience test advanced scientific reasoning. For BMB, we use the human-difficult subset, which requires analysis of real biological data. Training-stage and model-size comparisons use a separate 130-question text-only HLE subset (*HLE-130*); these scores should not be compared directly with the main HLE results in Table 1.

**Evaluation protocol:** For models evaluated in our harness, including GPT-5.6 Sol [28] and Gemini 3.1 Pro [12], we conduct three runs per benchmark, except BMB, for which we conduct five. We report the best score from a single run, rather than an aggregate over runs, and use each benchmark’s grading procedure. Reported score improvements are absolute differences between percentage scores, not relative changes. BMB evaluations allow access only to benchmark-approved websites. Baseline results are drawn from official reports, prior studies, or our evaluations, as distinguished in Table 1. Because externally reported baselines may differ in tool access or execution settings, cross-source comparisons are indicative rather than strictly controlled.

### 4.2 Overall Agentic Performance

Table 1 compares Axion 1.0 with open-weight and proprietary models across the six benchmarks. On full HLE, Axion 1.0-35B and Axion 1.0-122B achieve 52.2% and 59.0%, respectively, with access to websites such as Hugging Face blocked. Without these access restrictions, the corresponding scores are 59.1% and 64.8%.

Compared with the similarly sized Agents-A1, Axion 1.0-35B improves by 4.6% on HLE (52.2% vs. 47.6%), 5.8% on SEAL-0 (62.2% vs. 56.4%), and 7.0% on FrontierScience Olympiad (86.0% vs. 79.0%). On FrontierScience Research, the 35B and 122B models score 56.7% and 66.7%, respectively, compared with 40.0% for Agents-A1 and 26.7% for GPT-5.5; the 122B model also exceeds GPT-5.6 Sol’s 63.3% by 3.4%. The corresponding average rubric scores for our models are 6.67 and 7.09. On BMB, Axion 1.0-122B reaches 41.7%, compared with 47.1% for GPT-5.6 Sol and 46.1% for Claude Fable 5.

Increasing the model size from 35B to 122B improves five of the six benchmark scores. The largest gains are on FrontierScience Research (+10.0%), HLE (+6.8%), and BMB (+6.4%), with a smaller gain on HiPhO (+0.5%). SEAL-0 is the exception, with 62.2% for the 35B model and 58.6% for the 122B model. Overall, the results show strong performance across both scientific reasoning and information-seeking tasks, while also highlighting variation across benchmarks.

| Model                          | Humanity's<br>Last Exam | BioMysteryBench<br>(human-difficult) | SEAL-0            | HiPhO             | FrontierScience<br>Olympiad | FrontierScience<br>Research |
|--------------------------------|-------------------------|--------------------------------------|-------------------|-------------------|-----------------------------|-----------------------------|
| <i>Open-weight models</i>      |                         |                                      |                   |                   |                             |                             |
| Agents-A1 [7]                  | 47.6                    | —                                    | 56.4              | 46.4              | 79.0                        | 40.0                        |
| Kimi-K2.6 [23]                 | 54.0                    | —                                    | 50.5 <sup>†</sup> | 41.1 <sup>†</sup> | 73.0 <sup>†</sup>           | 17.9 <sup>†</sup>           |
| DeepSeek-V4-Pro (Preview) [10] | 48.2                    | —                                    | 55.0 <sup>†</sup> | 38.7 <sup>†</sup> | 76.0 <sup>†</sup>           | 13.3 <sup>†</sup>           |
| Apodex 1.1 (ReAct) [1]         | 53.2                    | 23.5                                 | —                 | —                 | —                           | 55.0                        |
| Apodex 1.1 (Agent Team) [1]    | 56.1                    | 35.3                                 | —                 | —                 | —                           | 63.3                        |
| <i>Proprietary models</i>      |                         |                                      |                   |                   |                             |                             |
| GPT-5.5 [27]                   | 52.2                    | —                                    | 42.3 <sup>†</sup> | 43.3 <sup>†</sup> | 78.0 <sup>†</sup>           | 26.7 <sup>†</sup>           |
| GPT-5.6 Sol [28]               | —                       | 47.1*                                | 53.2*             | 42.9*             | 79.0*                       | 63.3*                       |
| GPT-6 Astra [29]               | 57.2                    | —                                    | —                 | —                 | —                           | —                           |
| Muse-Spark-1.0 [21]            | 58.0                    | —                                    | —                 | —                 | —                           | 38.0                        |
| Gemini 3.1 Pro [12]            | 51.4                    | 35.3*                                | 53.2*             | 38.5*             | 85.0*                       | 26.7*                       |
| Claude Opus 5 [5]              | 63.6                    | 49.4                                 | —                 | —                 | —                           | —                           |
| Claude Fable 5 [4]             | 63.8                    | 46.1                                 | —                 | —                 | —                           | —                           |
| <b>Axion 1.0-35B</b>           | 52.2                    | 35.3                                 | 62.2              | 45.7              | 86.0                        | 56.7                        |
| <b>Axion 1.0-122B</b>          | 59.0                    | 41.7                                 | 58.6              | 46.2              | 87.0                        | 66.7                        |

**Table 1 Overall performance of Axion 1.0.** Benchmark scores are reported as percentages. Shaded rows indicate our models, and — indicates an unreported result. Axion 1.0 scores are evaluated by us. For baselines, <sup>†</sup> indicates evaluation by Agents-A1 or Apodex rather than the model provider, while \* indicates evaluation by us; unmarked scores are from official reports. Cross-source comparisons may involve different execution settings. Axion’s HLE scores use restricted website access; results without these restrictions are reported in the text. FrontierScience Research rubric scores are reported separately in the text.

### 4.3 Contribution of Each Training Stage

Table 2 follows each main model from its base checkpoint through SFT Stage I, SFT Stage II, and RL. These sequential checkpoint comparisons show how performance changes as training progresses; they are not component-removal ablations.

**Stage I: self-contained reasoning:** The first SFT stage yields its largest improvements on tasks requiring multi-step scientific reasoning. For the 35B model, HLE-130 increases from 47.7% to 59.2%, HiPhO from 38.6% to 45.9%, and FrontierScience Olympiad from 71.0% to 83.0%. FrontierScience Research also increases from 26.7% to 48.3%. The 122B model shows similar gains: HLE-130 increases from 40.8% to 68.5% and FrontierScience Olympiad from 65.0% to 85.0%. These improvements are consistent with Stage I strengthening the reasoning and computation emphasized during data construction. They are not universal, however: the 35B model’s BMB score decreases from 23.5% to 17.6% at this stage.

**Stage II: tool-dependent reasoning:** Adding tool-use SFT produces particularly clear gains on tasks involving external information and data analysis. For the 35B model, BMB increases from 17.6% to 35.3%, FrontierScience Research from 48.3% to 58.3%, SEAL-0 from 50.5% to 53.2%, and HLE-130 from 59.2% to 66.2%. For the 122B model, BMB rises from 29.4% to 35.3% and FrontierScience Research from 56.7% to 65.0%, while HLE-130 remains at 68.5%. These results support the intended progression from self-contained reasoning to finding, integrating, and computing with external information.

**Reinforcement learning:** RL further improves HLE-130 and SEAL-0 for both model sizes. For the 35B model, scores increase from 66.2% to 70.0% on HLE-130 and from 53.2% to 62.2% on SEAL-0; for the 122B model, they increase from 68.5% to 74.6% and from 53.2% to 58.6%, respectively. RL also improves BMB for the 122B model (35.3% to 41.7%). Its effects on scientific reasoning benchmarks vary by model size: FrontierScience Research declines from 58.3% to 56.7% for 35B but improves from 65.0% to 66.7% for 122B. HiPhO decreases slightly for 35B (46.8% to 45.7%) but improves slightly for 122B (45.8% to 46.2%). Thus, RL provides additional gains on agentic tasks, while its benefits on scientific reasoning benchmarks are more

| Training stage             | HLE-130 | BioMysteryBench<br>(human-difficult) | SEAL-0 | HiPhO | FrontierScience<br>Olympiad | FrontierScience<br>Research |
|----------------------------|---------|--------------------------------------|--------|-------|-----------------------------|-----------------------------|
| <b>Axion 1.0-35B-A3B</b>   |         |                                      |        |       |                             |                             |
| Qwen3.6-35B-A3B            | 47.7    | 23.5                                 | 44.1   | 38.6  | 71.0                        | 26.7                        |
| + SFT Stage I              | 59.2    | 17.6                                 | 50.5   | 45.9  | 83.0                        | 48.3                        |
| + SFT Stage II             | 66.2    | 35.3                                 | 53.2   | 46.8  | 84.0                        | 58.3                        |
| + RL (Axion 1.0)           | 70.0    | 35.3                                 | 62.2   | 45.7  | 86.0                        | 56.7                        |
| <b>Axion 1.0-122B-A10B</b> |         |                                      |        |       |                             |                             |
| Qwen3.5-122B-A10B          | 40.8    | 17.6                                 | 47.8   | 30.6  | 65.0                        | 40.0                        |
| + SFT Stage I              | 68.5    | 29.4                                 | 52.3   | 45.6  | 85.0                        | 56.7                        |
| + SFT Stage II             | 68.5    | 35.3                                 | 53.2   | 45.8  | 86.0                        | 65.0                        |
| + RL (Axion 1.0)           | 74.6    | 41.7                                 | 58.6   | 46.2  | 87.0                        | 66.7                        |

**Table 2 Performance across successive training stages.** Each block tracks one base model through SFT Stage I, SFT Stage II, and RL; shaded rows correspond to the final Axion 1.0 models. HLE uses the 130-question text-only subset. Scores are reported as percentages and follow the protocol in Section 4.1.

consistent for the larger model.

Overall, the checkpoint results indicate that much of the capability gain arises during the two SFT stages, with RL providing further improvements on selected benchmarks. This performance is achieved without dedicated mid-training, DPO, or OPD, consistent with the simple training recipe described in Section 3. Appendix B complements these results with selected trajectory comparisons illustrating self-checking after Stage I, evidence-grounded retrieval after Stage II, and the synthesis of conflicting information after RL.

#### 4.4 Additional Analysis

**Effect of the base model:** Figure 3 compares Qwen3.5-35B-A3B and Qwen3.6-35B-A3B before and after the same two SFT stages. Qwen3.5 initially trails Qwen3.6 on five of the six benchmarks, but the gap narrows substantially after SFT. On HLE-130, the difference decreases from 9.2% to 0.8%; on BMB, both models reach 35.3%. The relationship is not identical on every benchmark: after SFT, Qwen3.5 obtains a higher SEAL-0 score (57.7% vs. 53.2%). These results suggest that the two-stage SFT recipe is effective for both initializations, rather than relying exclusively on the stronger starting checkpoint.

**Effect of SFT across model scale:** Table 3 reports results for 4B, 9B, 35B, and 122B models. Among the smaller models, which receive the same two SFT stages without RL, the 9B model outperforms the 4B model on all six benchmarks. In particular, HLE-130 improves from 53.1% to 63.1%, BMB from 11.8% to 23.5%, SEAL-0 from 56.8% to 61.3%, and FrontierScience Research from 45.0% to 55.0%. This indicates that the constructed SFT data are effective at both smaller model sizes. The 35B and 122B results provide a broader view of final-model performance, but should not be interpreted as a controlled size-scaling experiment because these models additionally undergo RL and use different base architectures.

**Effect of RL across model scales:** Table 2 compares the 35B and 122B models before and after RL. For **Axion 1.0-35B**, RL yields its largest gains on SEAL-0 (+9.0%) and HLE-130 (+3.8%), with a smaller improvement on FrontierScience Olympiad (+2.0%). BMB remains unchanged, while performance decreases slightly on full HiPhO (−1.1%) and FrontierScience Research (−1.6%). For **Axion 1.0-122B**, RL yields its largest gains on BMB (+6.4%), HLE-130 (+6.1%), and SEAL-0 (+5.4%), with smaller improvements on full HiPhO and both FrontierScience subsets. Across these two models, RL consistently improves HLE-130, SEAL-0, and FrontierScience Olympiad, while its effects on the remaining benchmarks vary. These comparisons describe the gains for each model under its training configuration, rather than isolating model scale as their cause.

**Effect of rollout routing replay:** On Qwen3.5-122B-A10B, rollout routing replay (R3) [20] improves



**Figure 3 Effect of the base model.** Two 35B-A3B initializations are evaluated before and after the same two-stage SFT recipe. HLE results use the 130-question text-only subset. All scores follow the protocol in Section 4.1.

| Model          | HLE-130 | BioMysteryBench<br>(human-difficult) | SEAL-0 | HiPhO | FrontierScience<br>Olympiad | FrontierScience<br>Research |
|----------------|---------|--------------------------------------|--------|-------|-----------------------------|-----------------------------|
| Axion 1.0-4B   | 53.1    | 11.8                                 | 56.8   | 42.3  | 81.0                        | 45.0                        |
| Axion 1.0-9B   | 63.1    | 23.5                                 | 61.3   | 43.5  | 83.0                        | 55.0                        |
| Axion 1.0-35B  | 70.0    | 35.3                                 | 62.2   | 45.7  | 86.0                        | 56.7                        |
| Axion 1.0-122B | 74.6    | 41.7                                 | 58.6   | 46.2  | 87.0                        | 66.7                        |

**Table 3 Performance across model sizes.** The 4B and 9B models are trained with two SFT stages, while the 35B-A3B and 122B-A10B models additionally undergo RL. HLE uses the 130-question text-only subset. Scores are reported as percentages and follow the protocol in Section 4.1.

rollout-trainer consistency: the mean sequence-average log-probability gap decreases from 0.044 to 0.042, and the maximum sequence-average gap decreases from 1.799 to 1.282; mean gradient norm decreases from 1.998 to 1.922, mean PPO KL from 0.0413 to 0.0396, and mean clipping fraction from 3.86% to 3.68%. These improvements accompany stronger performance on HLE-130 (74.6% vs. 72.3%), FrontierScience Olympiad (87.0% vs. 85.0%), and FrontierScience Research (66.7% vs. 61.7%). HiPhO performance remains similar (46.2% vs. 46.3%), whereas SEAL-0 decreases from 63.1% to 58.6%. Overall, R3 improves rollout-trainer consistency and yields gains on several reasoning-heavy benchmarks, although the downstream benefits vary by benchmark. These results provide encouraging evidence for routing replay. However, the log-probability diagnostics also reflect asynchronous policy staleness and the off-policy effects of context condensation, limiting causal attribution of the benchmark differences to R3 alone.

## 5 Training Infrastructure

We develop separate infrastructure for full-parameter SFT on long agent trajectories and asynchronous RL with tool-interacting rollouts. Both are designed to train large mixture-of-experts (MoE) models efficiently

while preserving the interactions needed for agentic learning.

## 5.1 Supervised Fine-Tuning Framework

Our SFT framework integrates NVIDIA Megatron-FSDP [52] into the Megatron backend of ms-swift [53]. It combines ZeRO-3-style sharding of parameters, gradients, and optimizer states [33] with Megatron’s optimized kernels and parallelism [39]. Unlike session-based training with context condensation, SFT uses each complete agent trajectory as one example, preserving all conversation turns and tool observations without replacing earlier context with summaries. This retains the original interaction history but requires training on sequences approaching 128K tokens.

**Parallelism for large MoE models:** We use fully sharded data parallelism (FSDP) rather than pipeline parallelism [24], avoiding pipeline bubbles and stage-to-stage communication that otherwise require carefully tuned schedules [31]. FSDP distributes model states across GPUs and overlaps communication with computation. Within MoE layers, we use expert parallelism [18] instead of tensor parallelism [39]: keeping expert computations intact avoids splitting small expert matrix multiplications and introducing additional all-reduces. We set the tensor-parallel degree to 1 and the expert-parallel degree to 8, keeping expert dispatch and combination within a single NVLink domain.

**Context parallelism for long trajectories:** FSDP alone does not shard all activations. For Qwen3.5-122B-A10B, the remaining activation memory exceeds 1.4 MiB per token, including MoE dispatch buffers, gated-delta-net (GDN) states, and vocabulary logits. We therefore partition each sequence across GPUs, using Ring Attention [19] for full-attention layers and a Ulysses-style all-to-all [14] for GDN layers. On 128K-token sequences, context-parallel degree 4 and micro-batch size 2 provide the highest throughput in our tests.

**Training cost:** All SFT runs use a 128K context window on 32 GPUs. For Qwen3.5-122B-A10B, Stage I processes 841M tokens in 3 h 32 min and Stage II takes 3 h 58 min, totaling 7 h 30 min (240 GPU-hours), with peak memory of 113 GiB per GPU. For Qwen3.6-35B-A3B, Stage I processes 657M tokens in 1 h 2 min and Stage II takes 1 h 46 min, totaling 2 h 48 min (approximately 90 GPU-hours), with peak memory of 101 GiB per GPU.

## 5.2 Reinforcement Learning Framework

Our RL infrastructure builds on verl’s fully asynchronous policy module and separates rollout generation, policy optimization, and auxiliary services onto distinct GPU pools [37]. Rollout workers use vLLM to execute the agent loop with search, webpage visits, Python, and QUEST-based context condensation. Trainer workers run the Megatron-Core actor and reference model. A separate service node hosts two Qwen3.8-27B judges for open-ended and citation rewards and six Qwen3.5-9B instances for webpage summarization and context condensation. For Qwen3.5-122B-A10B, we use ten tensor-parallel-4 rollout engines on 40 GPUs and a 16-GPU trainer (tensor parallelism 2, expert parallelism 8, with optimizer states offloaded to CPU), alongside the service node.

**Asynchronous rollout and optimization:** Rollout and training workers run concurrently on separate GPUs. Rollout engines continuously generate trajectories and enqueue completed results; the trainer starts an update once it has a batch of 24 prompts with non-degenerate rewards. Every two optimizer steps, updated weights are transferred to the rollout engines through NCCL, taking 45–60 seconds for the 122B model. In-flight rollouts continue with the updated weights, while trajectories more than 11 parameter versions old are discarded. This separation avoids the inference-memory contention of colocating rollout and training workers and improves throughput for the 122B model. Following QUEST, context-condensation boundaries divide RL rollouts into sessions. Sessions from the same rollout share its reward and group-normalized advantage, and the training loss is normalized per session.

**Token consistency with Token-In-Token-Out (TITO):** Reconstructing sessions by rendering messages and re-tokenizing them can change token IDs because byte-pair encoding (BPE) is sensitive to token boundaries. This creates a mismatch between the sequence used for rollout and the one used for training. We avoid this with TITO [37]: the agent loop exchanges token IDs directly, appends generated and tool-output tokens to a

shared token buffer, and supplies the trainer with that buffer and a mask of policy-generated positions. After context condensation, the new prefix is tokenized only once at session start and joined to subsequent tokens without a decode-re-encode cycle.

**Expert-routing consistency with Rollout Routing Replay (R3):** For MoE models, vLLM and Megatron can select different experts for the same tokens even with identical weights. R3 addresses this mismatch by recording the selected experts for each token and MoE layer during rollout and replaying those decisions in the trainer’s forward pass [20]. The routing records use `uint8` tensors (approximately 20 MB per sample). On Qwen3.5-122B-A10B, R3 reduces mean gradient norm from 1.998 to 1.922, mean PPO KL from 0.0413 to 0.0396, and mean clipping fraction from 3.86% to 3.68%. It also halves the number of intervals with gradient norm above 5. However, the largest gradient spike increases from 21.1 to 26.8 and reward standard deviation rises by 15%, indicating improved average consistency rather than elimination of occasional instability.

## 6 Related Work

**Agent foundation models:** Recent models increasingly integrate reasoning, coding, and tool use as general-purpose capabilities, as illustrated by Kimi K2 [16] and GLM-4.5 [11]. Research-oriented agents extend this direction to sustained information gathering and interaction with external environments. MiroThinker-1.7 [22] combines agentic mid-training, SFT, preference optimization, and RL; Agents-A1 [7] uses long-horizon supervision and domain-routed on-policy distillation; and Apodex 1.1 [1] investigates executable environments and agent coordination. These systems explore different approaches to building capable agents. In contrast, Axion 1.0 studies how far a simpler sequence of two SFT stages and RL can go, without dedicated mid-training, direct preference optimization, or on-policy distillation.

**Construction of challenging and verifiable tasks:** Synthetic data has long been used to expand training supervision. Self-Instruct [42] generates instruction data from model outputs, while Evol-Instruct [46] increases task complexity through iterative transformations. More recent work adapts task difficulty to the learner: Absolute Zero [51] uses execution-grounded self-play, and Agent0 [44] develops curricula through interaction between a task generator and a tool-using agent. Axion’s first data stage similarly uses model feedback, but specifically strengthens questions that the base model solves correctly by targeting its successful solution strategy. This produces harder, self-contained reasoning tasks with independently verified answers.

Tool-use training additionally requires tasks that demand external information while remaining verifiable. TaskCraft [38] expands verifiable multi-tool tasks along depth and breadth, WebShaper [40] structures information-seeking complexity, and QUEST [45] constructs research tasks with verifiable rubrics. Axion’s second data stage takes a complementary approach: deterministic filtering, unique-entity selection, and computation over structured data produce reference answers before an LLM formulates the questions. With construction-time data withheld and closed-book-solvable questions filtered out, the resulting tasks emphasize retrieval, webpage inspection, and grounded computation.

**Reinforcement learning for reasoning and tool use:** DeepSeek-R1 [9] demonstrates the potential of RL to improve reasoning. Search-R1 [15] extends RL to reasoning interleaved with retrieval, DeepResearcher [55] trains agents through open-web interaction, and AgentGym-RL [43] studies multi-turn learning across environments. Together, these works show how RL can improve sequential decisions, not only final responses. Axion applies RL after its two complementary SFT stages, building on supervised reasoning and tool-use trajectories to improve long-horizon interaction.

**Training–inference consistency in RL:** Agentic RL typically generates trajectories with an inference engine and computes policy updates in a separate training backend. Differences in numerical execution, expert routing, or tokenization can cause the two systems to assign different probabilities to the same trajectory, complicating policy optimization. One line of work addresses this mismatch through truncated importance sampling [47], sequence-aware masking [17], and analyses of the conditions under which token-level corrections remain valid [54]. Another seeks to reduce the discrepancy itself through expert-routing replay [20], numerical-format choices [32], and precision-aware optimization [50]. Controlled studies further examine the

instability caused by numerical mismatch [56]. In long agent trajectories, reconstructing messages and tool observations can introduce an additional tokenization mismatch. Axion addresses these two sources directly: Token-In-Token-Out (TITO) preserves the rollout token sequence across training, while Rollout Routing Replay (R3) aligns MoE expert choices between the inference and training engines.

**Open and reproducible training recipes:** Reproducibility requires more than releasing model weights: data sources, construction procedures, and training configurations also determine the resulting capabilities. OpenThoughts [13] develops open reasoning-data recipes, and OpenThoughts-Agent [34] investigates and releases agentic data-curation procedures. DR Tulu [36] provides an open RL framework for research agents, while QUEST [45] releases models, data, and training scripts. Axion follows this direction by making its three-stage recipe—including data generation, filtering, verification, prompts, and training procedures—available for inspection and reproduction. Its emphasis is on obtaining broad agentic capabilities with a compact, reproducible pipeline.

## 7 Conclusion

We present Axion 1.0, demonstrating that strong agentic capabilities can be developed through a simple yet effective, data-centric training paradigm. Our approach progressively equips models with the ability to reason over complex problems, acquire and utilize external knowledge, and refine their capabilities through reinforcement learning. Across challenging evaluations, Axion 1.0 demonstrates strong scientific reasoning and tool-use capabilities, highlighting the effectiveness of this streamlined approach. More broadly, our findings suggest that advancing agent foundation models does not necessarily require increasingly complex training pipelines; carefully constructed data and well-designed learning objectives can be equally powerful. By releasing our data construction pipelines and training code, we hope to provide a reproducible and accessible foundation for future research toward more capable and general-purpose AI agents.

## References

- [1] B. An, B. Li, B. Wang, B. Zhang, B. L. Wang, C. Feng, C. Wei, C. Xue, C. Zhang, D. Ng, D. Ye, E. Min, F. Chen, F. Liu, F. Yang, F. Ye, G. Sun, H. Ji, H. Xu, H. Yang, H. Ye, H. Zhang, H. Zhao, J. Li, J. Lin, J. Xia, K. Jin, K. Wang, K. Yang, L. Bing, L. Lei, L. Su, Le. Wang, Lu. Wang, N. Wang, Q. Ren, Q. Yang, R. Li, S. Bai, S. Du, S. Li, S. Lin, S. Nie, S. Wang, S. Zhang, S. Z. Wang, T. Ge, Ta. Q. Fang, Ti. Q. Fang, W. Fang, W. Li, W. Zhang, X. Chen, X. Li, X. Tang, X. Wang, X. Xu, X. Zhang, X. Q. Wang, X. Y. Wang, Y. Deng, Y. Gao, Y. Hu, Y. Li, Y. Sui, Y. Wang, Y. Xiao, Y. Zhang, Y. Zhou, Z. Chen, Z. Cheng, Z. Feng, Z. Liang, Z. Liu, and Z. Zhang. Apodex 1.1: Scaling agentic intelligence for complex work. *arXiv preprint arXiv:2608.23283*, 2026. URL <https://arxiv.org/abs/2608.23283>.
- [2] Anthropic. System card: Claude Opus 4 and Claude Sonnet 4, may 2025. URL <https://www-cdn.anthropic.com/4263b940cabb546aa0e3283f35b686f4f3b2ff47.pdf>.
- [3] Anthropic. Evaluating Claude’s bioinformatics research capabilities with BioMysteryBench, April 2026. URL <https://www.anthropic.com/research/Evaluating-Claude-For-Bioinformatics-With-BioMysteryBench>.
- [4] Anthropic. Claude Fable 5 and Claude Mythos 5 system card, jun 2026. URL <https://anthropic.com/claude-fable-5-mythos-5-system-card>.
- [5] Anthropic. Claude Opus 5 system card, jul 2026. URL <https://anthropic.com/claude-opus-5-system-card>.
- [6] Anthropic. Claude Opus 5.5 system card, sep 2026. URL <https://www.anthropic.com/claude-opus-5-5-system-card>.
- [7] Lei Bai, Zongsheng Cao, Yang Chen, Zhiyao Cui, Shangheng Du, Yue Fan, Shiyang Feng, Zijie Guo, Haonan He, Liang He, Xiaohan He, Shuyue Hu, Yusong Hu, Songtao Huang, Yichen Jiang, Hao Li, Xin Li, Dahua Lin, Weihao Lin, Fenghua Ling, Dongrui Liu, Zhuo Liu, Wenjie Lou, Runmin Ma, Chunjiang Mu, Haoyang Peng, Tianshuo Peng, Jinxin Shi, Luohe Shi, Boyuan Sun, Zelin Tan, Shengji Tang, Yan Teng, Qianyi Wang, Xiaosong Wang, Yiming Wu, Yi Xie, Xiangchao Yan, Jingqi Ye, Peng Ye, Fangchen Yu, Jiakang Yuan, Bihao Zhan, Bo Zhang, Chen Zhang, Shufei Zhang, Shuaiyu Zhang, Wenlong Zhang, Yiqun Zhang, Junpeng Zhao, Zhijie Zhong, Bowen Zhou, and Yuhao Zhou. Scaling the horizon, not the parameters: Reaching trillion-parameter performance with a 35B agent. *arXiv preprint arXiv:2606.30616*, 2026. URL <https://arxiv.org/abs/2606.30616>.
- [8] Center for AI Safety, Scale AI, and HLE Contributors Consortium. A benchmark of expert-level academic questions to assess AI capabilities. *Nature*, 649:1139–1146, 2026. doi: 10.1038/s41586-025-09962-4. URL <https://arxiv.org/abs/2501.14249>.
- [9] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojuan Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanbiao Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yudian Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang,

- and Zhen Zhang. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL <https://arxiv.org/abs/2501.12948>.
- [10] DeepSeek-AI et al. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL <https://arxiv.org/abs/2606.19348>.
- [11] GLM-4.5 Team, 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, Yifan An, Yilin Niu, Yuanhao Wen, Yushi Bai, Zhengxiao Du, Zihan Wang, Zilin Zhu, Bohan Zhang, Bosi Wen, Bowen Wu, Bowen Xu, Can Huang, Casey Zhao, Changpeng Cai, Chao Yu, Chen Li, Chendi Ge, Chenghua Huang, Chenhui Zhang, Chenxi Xu, Chenzheng Zhu, Chuang Li, Congfeng Yin, Daoyan Lin, Dayong Yang, Dazhi Jiang, Ding Ai, Erle Zhu, Fei Wang, Gengzheng Pan, Guo Wang, Hailong Sun, Haitao Li, Haiyang Li, Haiyi Hu, Hanyu Zhang, Hao Peng, Hao Tai, Haoke Zhang, Haoran Wang, Haoyu Yang, He Liu, He Zhao, Hongwei Liu, Hongxi Yan, Huan Liu, Huilong Chen, Ji Li, Jiajing Zhao, Jiamin Ren, Jian Jiao, Jiani Zhao, Jianyang Yan, Jiaqi Wang, Jiayi Gui, Jiayue Zhao, Jie Liu, Jijie Li, Jing Li, Jing Lu, Jingsen Wang, Jingwei Yuan, Jingxuan Li, Jingzhao Du, Jinhua Du, Jinxin Liu, Junkai Zhi, Junli Gao, Ke Wang, Lekang Yang, Liang Xu, Lin Fan, Lindong Wu, Lintao Ding, Lu Wang, Man Zhang, Minghao Li, Minghuan Xu, Mingming Zhao, Mingshu Zhai, Pengfan Du, Qian Dong, Shangde Lei, Shangqing Tu, Shangtong Yang, Shaoyou Lu, Shijie Li, Shuang Li, Shuang-Li, Shuxun Yang, Sibao Yi, Tianshu Yu, Wei Tian, Weihang Wang, Wenbo Yu, Weng Lam Tam, Wenjie Liang, Wentao Liu, Xiao Wang, Xiaohan Jia, Xiaotao Gu, Xiaoying Ling, Xin Wang, Xing Fan, Xingru Pan, Xinyuan Zhang, Xinze Zhang, Xiuqing Fu, Xunkai Zhang, Yabo Xu, Yandong Wu, Yida Lu, Yidong Wang, Yilin Zhou, Yiming Pan, Ying Zhang, Yingli Wang, Yingru Li, Yinpei Su, Yipeng Geng, Yitong Zhu, Yongkun Yang, Yuhang Li, Yuhao Wu, Yujiang Li, Yunan Liu, Yuning Wang, Yuntao Li, Yuxuan Zhang, Zezhen Liu, Zhen Yang, Zhengda Zhou, Zhongpei Qiao, Zhuoer Feng, Zhuorui Liu, Zichen Zhang, Zihan Wang, Zijun Yao, Zikang Wang, Ziqiang Liu, Ziwei Chai, Zixuan Li, Zuodong Zhao, Wenguang Chen, Jidong Zhai, Bin Xu, Minlie Huang, Hongning Wang, Juanzi Li, Yuxiao Dong, and Jie Tang. GLM-4.5: Agentic, reasoning, and coding (ARC) foundation models. *arXiv preprint arXiv:2508.06471*, 2025. URL <https://arxiv.org/abs/2508.06471>.
- [12] Google DeepMind. Gemini 3.1 Pro model card, feb 2026. URL <https://deepmind.google/models/model-cards/gemini-3-1-pro>.
- [13] Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanxia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. OpenThoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*, 2025. URL <https://arxiv.org/abs/2506.04178>.
- [14] Sam Ade Jacobs, Masahiro Tanaka, Chengming Zhang, Minjia Zhang, Shuaiwen Leon Song, Samyam Rajbhandari, and Yuxiong He. DeepSpeed ulyssees: System optimizations for enabling training of extreme long sequence transformer models, 2023. URL <https://arxiv.org/abs/2309.14509>.
- [15] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025. URL <https://arxiv.org/abs/2503.09516>.
- [16] Kimi Team, Yifan Bai, Yiping Bao, Y. Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong 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, Kelin Fu, Bofei Gao, Chenxiao Gao, Hongcheng Gao, Peizhong Gao, Tong Gao, Yuyao Ge, Shangyi Geng, Qizheng Gu, Xinran Gu, Longyu Guan, Haiqing Guo, Jianhang Guo, Xiaoru Hao, Tianhong He, Weiran He, Wenyang He, Yunjia He, Chao Hong, Hao Hu, Yangyang Hu, Zhenxing Hu, Weixiao Huang, Zhiqi Huang, Zihao Huang, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yongsheng Kang, Guokun Lai, Cheng Li, Fang Li, Haoyang Li, Ming Li, Wentao Li, Yang Li, Yanhao Li, Yiwei Li, Zhaowei Li, Zheming Li, Hongzhan Lin, Xiaohan Lin, Zongyu Lin, Chengyin Liu, Chenyu Liu, Hongzhang Liu, Jingyuan Liu, Junqi Liu, Liang Liu, Shaowei Liu, T. Y. Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yibo Liu, Yiping Liu, Yue Liu, Zhengying Liu, Enzhe Lu, Haoyu Lu, Lijun Lu, Yashuo Luo, Shengling Ma, Xinyu Ma, Yingwei Ma, Shaoguang Mao, Jie Mei, Xin Men, Yibo Miao, Siyuan Pan, Yebo Peng, Ruoyu Qin, Zeyu Qin, Bowen Qu, Zeyu Shang, Lidong Shi, Shengyuan Shi, Feifan Song, Jianlin Su,

- Zhengyuan Su, Lin Sui, Xinjie Sun, Flood Sung, Yunpeng Tai, Heyi Tang, Jiawen Tao, Qifeng Teng, Chaoran Tian, Chensi Wang, Dinglu Wang, Feng Wang, Hailong Wang, Haiming Wang, Jianzhou Wang, Jiaxing Wang, Jinhong Wang, Shengjie Wang, Shuyi Wang, Si Wang, Xinyuan Wang, Yao Wang, Yejie Wang, Yiqin Wang, Yuxin Wang, Yuzhi Wang, Zhaoji Wang, Zhengtao Wang, Zhengtao Wang, Zhexu Wang, Chu Wei, Qianqian Wei, Haoning Wu, Wenhao Wu, Xingzhe Wu, Yuxin Wu, Chenjun Xiao, Jin Xie, Xiaotong Xie, Weimin Xiong, Boyu Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ziyao Xu, Jing Xu, Jing Xu, Junjie Yan, Yuzi Yan, Hao Yang, Xiaofei Yang, Yi Yang, Ying Yang, Zhen Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Xingcheng Yao, Wenjie Ye, Zhuorui Ye, Bohong Yin, Longhui Yu, Enming Yuan, Hongbang Yuan, Mengjie Yuan, Siyu Yuan, Haobing Zhan, Dehao Zhang, Hao Zhang, Wanlu Zhang, Xiaobin Zhang, Yadong Zhang, Yangkun Zhang, Yichi Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Haotian Zhao, Yikai Zhao, Zijia Zhao, Huabin Zheng, Shaojie Zheng, Longguang Zhong, Jianren Zhou, Xinyu Zhou, Zaida Zhou, Jinguo Zhu, Zhen Zhu, Weiyu Zhuang, and Xinxing Zu. Kimi K2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025. URL <https://arxiv.org/abs/2507.20534>.
- [17] Yingru Li, Jiakai Liu, Jiawei Xu, Yuxuan Tong, Ziniu Li, Qian Liu, and Baoxiang Wang. Trust region masking for long-horizon llm reinforcement learning, 2026. URL <https://arxiv.org/abs/2512.23075>.
- [18] Dennis Liu, Zijie Yan, Xin Yao, Tong Liu, Vijay Korthikanti, Evan Wu, Shiqing Fan, Gao Deng, Hongxiao Bai, Jianbin Chang, Ashwath Aithal, Michael Andersch, Mohammad Shoeybi, Jiajie Yao, Chandler Zhou, David Wu, Xipeng Li, and June Yang. Moe parallel folding: Heterogeneous parallelism mappings for efficient large-scale moe model training with megatron core, 2026. URL <https://arxiv.org/abs/2504.14960>.
- [19] Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context, 2023. URL <https://arxiv.org/abs/2310.01889>.
- [20] Wenhan Ma, Hailin Zhang, Liang Zhao, Yifan Song, Yudong Wang, Zhifang Sui, and Fuli Luo. Stabilizing moe reinforcement learning by aligning training and inference routers, 2025. URL <https://arxiv.org/abs/2510.11370>.
- [21] Meta. Introducing Muse Spark: Scaling towards personal superintelligence, apr 2026. URL <https://ai.meta.com/blog/introducing-muse-spark-1-msl/>.
- [22] MiroMind Team, S. Bai, L. Bing, L. Lei, R. Li, X. Li, X. Lin, E. Min, L. Su, B. Wang, L. Wang, L. Wang, S. Wang, X. Wang, Y. Zhang, Z. Zhang, G. Chen, L. Chen, Z. Cheng, Y. Deng, Z. Huang, D. Ng, J. Ni, Q. Ren, X. Tang, B. L. Wang, H. Wang, N. Wang, C. Wei, Q. Wu, J. Xia, Y. Xiao, H. Xu, X. Xu, C. Xue, Z. Yang, Z. Yang, F. Ye, H. Ye, J. Yu, C. Zhang, W. Zhang, H. Zhao, and P. Zhu. MiroThinker-1.7 & H1: Towards heavy-duty research agents via verification. *arXiv preprint arXiv:2603.15726*, 2026. URL <https://arxiv.org/abs/2603.15726>.
- [23] Moonshot AI. Kimi K2.6: Advancing open-source coding. Technical blog, 2026. URL <https://www.kimi.ai/blog/kimi-k2-6>.
- [24] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Anand Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. Efficient large-scale language model training on gpu clusters using megatron-lm, 2021. URL <https://arxiv.org/abs/2104.04473>.
- [25] OpenAI. ChatGPT agent system card, jul 2025. URL <https://openai.com/index/chatgpt-agent-system-card/>.
- [26] OpenAI. Evaluating AI’s ability to perform scientific research tasks, December 2025. URL <https://openai.com/index/frontierscience/>.
- [27] OpenAI. GPT-5.5 system card, apr 2026. URL <https://openai.com/index/gpt-5-5-system-card/>.
- [28] OpenAI. GPT-5.6 system card, jul 2026. URL <https://deploymentsafety.openai.com/gpt-5-6>.
- [29] OpenAI. GPT-6 Astra system card, sep 2026. URL <https://deploymentsafety.openai.com/gpt-6-astra>.
- [30] Thinh Pham, Nguyen Nguyen, Pratibha Zunjare, Weiyan Chen, Yu-Min Tseng, and Tu Vu. SealQA: Raising the bar for reasoning in search-augmented language models. *arXiv preprint arXiv:2506.01062*, 2025. URL <https://arxiv.org/abs/2506.01062>.
- [31] Penghui Qi, Xinyi Wan, Guangxing Huang, and Min Lin. Zero bubble pipeline parallelism, 2023. URL <https://arxiv.org/abs/2401.10241>.
- [32] Penghui Qi, Zichen Liu, Xiangxin Zhou, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Defeating the training-inference mismatch via fp16, 2025. URL <https://arxiv.org/abs/2510.26788>.

- [33] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models, 2020. URL <https://arxiv.org/abs/1910.02054>.
- [34] Negin Raoof, Richard Zhuang, Marianna Nezhurina, Etash Guha, Atula Tejaswi, Ryan Marten, Charlie F. Ruan, Tyler Griggs, Alexander Glenn Shaw, Hritik Bansal, E. Kelly Buchanan, Artem Gazizov, Reinhard Heckel, Chinmay Hegde, Sankalp Jajee, Daanish Khazi, Emmanouil Koukoumidis, Xiangyi Li, Hange Liu, Shlok Natarajan, Harsh Raj, Nicholas Roberts, Ethan Shen, Nishad Singhi, Michael Siu, Ashima Suvarna, Hanwen Xing, Patrick Yubeaton, Robert Zhang, Leon Liangyu Chen, Xiaokun Chen, Steven Dillmann, Saadia Gabriel, Xunyi Jiang, Anurag Kashyap, Boxuan Li, Yein Park, Minh Pham, Sujay Sanghavi, Lin Shi, Ke Sun, Yixin Wang, Zhiwei Xu, Erica Zhang, Siyan Zhao, Wanjia Zhao, Jenia Jitsev, Alex Dimakis, Benjamin Feuer, and Ludwig Schmidt. OpenThoughts-Agent: Data recipes for agentic models. *arXiv preprint arXiv:2606.24855*, 2026. URL <https://arxiv.org/abs/2606.24855>.
- [35] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.
- [36] Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, Varsha Kishore, Jingming Zhuo, Xinran Zhao, Molly Park, Samuel G. Finlayson, David Sontag, Tyler Murray, Sewon Min, Pradeep Dasigi, Luca Soldaini, Faeze Brahman, Wen tau Yih, Tongshuang Wu, Luke Zettlemoyer, Yoon Kim, Hannaneh Hajishirzi, and Pang Wei Koh. DR Tulu: Reinforcement learning with evolving rubrics for deep research. *arXiv preprint arXiv:2511.19399*, 2025. URL <https://arxiv.org/abs/2511.19399>.
- [37] Guangming Sheng, Chi Zhang, Zilinfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. *arXiv preprint arXiv:2409.19256*, 2024. URL <https://arxiv.org/abs/2409.19256>.
- [38] Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Liu, Yuchen Eleanor Jiang, Jian Yang, Ge Zhang, Jiaheng Liu, Changwang Zhang, Jun Wang, and Wangchunshu Zhou. TaskCraft: Automated generation of agentic tasks. In *The Fourteenth International Conference on Learning Representations*, 2026. URL [https://proceedings.iclr.cc/paper\\_files/paper/2026/hash/48644509339cb3076f7b0407c7588af6-Abstract-Conference.html](https://proceedings.iclr.cc/paper_files/paper/2026/hash/48644509339cb3076f7b0407c7588af6-Abstract-Conference.html).
- [39] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism, 2020. URL <https://arxiv.org/abs/1909.08053>.
- [40] Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. WebShaper: Agentially data synthesizing via information-seeking formalization. *arXiv preprint arXiv:2507.15061*, 2025. URL <https://arxiv.org/abs/2507.15061>.
- [41] 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.
- [42] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-Instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.754. URL <https://aclanthology.org/2023.acl-long.754/>.
- [43] Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Honglin Guo, Jiaqi Liu, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, Wei He, Yiwen Ding, Guanyu Li, Zehui Chen, Zhengyin Du, Xuesong Yao, Yufei Xu, Jiecao Chen, Tao Gui, Zuxuan Wu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. AgentGym-RL: Training LLM agents for long-horizon decision making through multi-turn reinforcement learning. *arXiv preprint arXiv:2509.08755*, 2025. URL <https://arxiv.org/abs/2509.08755>.
- [44] Peng Xia, Kaide Zeng, Jiaqi Liu, Can Qin, Fang Wu, Yiyang Zhou, Caiming Xiong, and Huaxiu Yao. Agent0: Unleashing self-evolving agents from zero data via tool-integrated reasoning. *arXiv preprint arXiv:2511.16043*, 2025. URL <https://arxiv.org/abs/2511.16043>.
- [45] Jian Xie, Tianhe Lin, Zilu Wang, Yuting Ning, Yuekun Yao, Tianci Xue, Zhehao Zhang, Zhongyang Li, Kai Zhang, Yufan Wu, Shijie Chen, Boyu Gou, Mingzhe Han, Yifei Wang, Vint Lee, Xinpeng Wei, Xiangjun Wang,

- Yu Su, and Huan Sun. QUEST: Training frontier deep research agents with fully synthetic tasks. *arXiv preprint arXiv:2605.24218*, 2026. URL <https://arxiv.org/abs/2605.24218>.
- [46] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=CfXh93NDgH>.
- [47] Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient RL framework secretly brings you off-policy RL training, 2025. URL <https://fengyao.notion.site/off-policy-rl>. Blog post.
- [48] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL [https://openreview.net/forum?id=WE\\_vluYUL-X](https://openreview.net/forum?id=WE_vluYUL-X).
- [49] Fangchen Yu, Haiyuan Wan, Qianjia Cheng, Yuchen Zhang, Jiacheng Chen, Fujun Han, Yulun Wu, Junchi Yao, Ruilizhen Hu, Ning Ding, Yu Cheng, Tao Chen, Lei Bai, Dongzhan Zhou, Yun Luo, Ganqu Cui, and Peng Ye. HiPhO: How far are (M)LLMs from humans in the latest high school physics olympiad benchmark? *arXiv preprint arXiv:2509.07894*, 2025. URL <https://arxiv.org/abs/2509.07894>.
- [50] Yaxiang Zhang, Yingru Li, Jiakai Liu, Jiawei Xu, Ziniu Li, Qian Liu, and Haoyuan Li. Beyond precision: Training-inference mismatch is an optimization problem and simple lr scheduling fixes it, 2026. URL <https://arxiv.org/abs/2602.01826>.
- [51] Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute Zero: Reinforced self-play reasoning with zero data. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3534. URL [https://papers.nips.cc/paper\\_files/paper/2025/hash/9837dc00ff67d176373268ed48042d49-Abstract-Conference.html](https://papers.nips.cc/paper_files/paper/2025/hash/9837dc00ff67d176373268ed48042d49-Abstract-Conference.html).
- [52] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. Pytorch fsdp: Experiences on scaling fully sharded data parallel, 2023. URL <https://arxiv.org/abs/2304.11277>.
- [53] Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. Swift:a scalable lightweight infrastructure for fine-tuning, 2024. URL <https://arxiv.org/abs/2408.05517>.
- [54] Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, Hao Lin, Chencan Wu, Feng Hu, An Yang, Jingren Zhou, and Junyang Lin. Stabilizing reinforcement learning with llms: Formulation and practices, 2025. URL <https://arxiv.org/abs/2512.01374>.
- [55] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deep-Researcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025. URL <https://arxiv.org/abs/2504.03160>.
- [56] Tianle Zhong, Neiwen Ling, Yifan Pi, Zijun Wei, Tianshu Yu, Geoffrey Fox, Peng Wu, and Xiao Yu. Diagnosing training inference mismatch in llm reinforcement learning via a zero-mismatch reference, 2026. URL <https://arxiv.org/abs/2605.14220>.

# Appendix

## A Training Hyperparameters

Tables 4 and 5 summarize the SFT and RL hyperparameters, respectively. Both SFT stages use the same optimization settings, with Stage II initialized from the Stage I checkpoint.

| Hyperparameter            | Value              |
|---------------------------|--------------------|
| Learning rate             | $3 \times 10^{-6}$ |
| Learning-rate schedule    | Constant           |
| Warmup steps              | 0                  |
| Adam $(\beta_1, \beta_2)$ | (0.9, 0.95)        |
| Weight decay              | 0.1                |
| Gradient clipping         | 1.0                |
| Training precision        | BF16               |
| Epochs per stage          | 3                  |
| Maximum sequence length   | 128K               |
| Global batch size         | 8                  |

**Table 4 SFT hyperparameters.** Settings apply to both supervised stages. Batch size counts sequences per optimizer update.

| Hyperparameter                    | Value              |
|-----------------------------------|--------------------|
| Learning rate                     | $2 \times 10^{-6}$ |
| Learning-rate schedule            | Constant           |
| Warmup steps                      | 5                  |
| Adam $(\beta_1, \beta_2)$         | (0.9, 0.999)       |
| Weight decay                      | 0.1                |
| Gradient clipping                 | 1.0                |
| Training precision                | BF16               |
| Rollouts per training question    | 8                  |
| Rollout temperature               | 1.0                |
| Maximum session prompt length     | 12,000             |
| Maximum session response length   | 49,152             |
| Lower / upper clipping range      | 0.2 / 0.28         |
| KL-loss coefficient               | 0.001              |
| Entropy coefficient               | 0                  |
| Parameter-sync interval (updates) | 2                  |
| Async staleness threshold         | 11                 |

**Table 5 RL hyperparameters.** The response-length limit applies to each session rather than the entire multi-session rollout. The eight rollouts used during training are separate from the four offline attempts used for task selection.

## B Case Studies Across Training Stages

We present three selected evaluation cases illustrating the progression from self-contained reasoning to evidence-grounded problem solving and the synthesis of conflicting information. Each case compares an earlier checkpoint with a later one on the same question. The examples complement the aggregate results in Table 2: they describe particular solution trajectories rather than establishing that a behavior is absent from every earlier-stage rollout or isolating the causal contribution of a training component.

| Transition                         | Task                      | Capability illustrated                                                                 |
|------------------------------------|---------------------------|----------------------------------------------------------------------------------------|
| Base $\rightarrow$ SFT Stage I     | Cube-surface geometry     | Detecting a flawed numerical approximation and revising the solution method.           |
| SFT Stage I $\rightarrow$ Stage II | Philosophical attribution | Retrieving a passage that directly supports the requested attribution.                 |
| SFT Stage II $\rightarrow$ RL      | Ride-booking application  | Separating competing usage metrics and resolving them against the question’s criteria. |

**Table 6 Overview of the selected case studies.** The comparisons concern selected 35B trajectories; the discussion below identifies the relevant behavior and answer in each case.

### B.1 Base to Stage I: Reasoning and Self-Checking

**Task: geodesic distance on a cube.** Let  $S$  be the surface of a cube of side length  $r$ , and let  $P$  be the midpoint of an edge. The locus  $C$  consists of points whose shortest distance to  $P$  along  $S$  is exactly  $r$ . Find  $|C|/(2\pi r)$  as a whole-number percentage.

**Earlier behavior:** The case-study record reports that Qwen3.6-35B-A3B attempted to unfold the cube, but mishandled the coordinate transformations and the assignment of circular arcs to faces. It obtained a total length of  $2\pi r$  and answered 100%, without checking the derivation computationally.

**Stage I behavior:** The selected Stage I trajectory first constructs a grid approximation and applies Dijkstra’s algorithm to estimate distances along the surface. It then tests this solver against a configuration with a known answer. For  $r = 1$ , take

$$P = (0, 0, 1/2), \quad Q = (0, 1/4, 1).$$

Both points lie on one face, so the straight segment on that face has length

$$d(P, Q) = \sqrt{(1/4)^2 + (1/2)^2} \approx 0.559.$$

The grid solver instead returns approximately 0.604, and grid refinement does not remove the discrepancy. The trajectory identifies the problem: movement restricted to grid edges produces a longer polyline rather than the straight surface path. It consequently switches to analytical face unfolding, compares alternative paths, and evaluates the resulting arcs. Its final expression is equivalent to

$$\frac{|C|}{2\pi r} = 1 - \frac{\arcsin(3/4)}{\pi} \approx 0.73005,$$

yielding the correct whole-number answer, 73%.

**Capability illustrated:** The distinguishing behavior is the use of a known geometric relation to challenge a computed result. Python supplies a candidate calculation, but the trajectory checks its validity, diagnoses a representation error, and changes method before committing to an answer.

## B.2 Stage I to Stage II: Retrieval and Use of Evidence

**Task: attributing a philosophical distinction.** Which philosopher of the Lviv–Warsaw School decomposes the notion of a hallmark into symptoms and signals?

**Earlier behavior:** In the selected 35B Stage I rollout, the model searches for material on the school’s theory of signs and attributes the distinction to its founder, Kazimierz Twardowski. The retrieved material is related to the topic, but the trajectory does not establish that this particular distinction belongs to him.

**Stage II behavior:** The selected 35B Stage II rollout searches for the terms of the distinction and opens material from *Studia Semiotyczne*. It locates a passage discussing Izydora Dąmbska’s treatment of indications, including their division into symptoms and signals. It connects this passage to the surrounding discussion of Dąmbska’s semiotic work and answers **Izydora Dmbska**.

**Capability illustrated:** Both checkpoints can invoke retrieval tools; the difference in these trajectories lies in how the evidence is used. Stage II connects the requested concept, the specific distinction, and the named philosopher, instead of treating general relevance to the school as sufficient support. This case illustrates evidence-grounded attribution rather than tool invocation alone.

## B.3 Stage II to RL: Reconciling Retrieved Information

**Task: identifying a ride-booking application.** Which ride-booking application is most widely used in India and offers a wide variety of vehicle choices and prices? The evaluation prompt asks for information applicable to 2024, when the question was created.

**Earlier behavior:** The selected 35B Stage II trajectory retrieves information about Ola, Uber, and Rapido. Its final response includes active-usage and ride-volume figures favoring Uber over Ola, but also emphasizes Ola’s cumulative installs and historical prominence. It presents Ola and Uber side by side and ultimately favors Ola, rather than resolving which evidence answers the combined criteria.

**RL behavior:** The selected 35B RL trajectory distinguishes cumulative downloads, active-user measures, ride counts, and vehicle-category coverage. It notes that the interpretation of “most widely used” depends on the metric and that reports can disagree. It then evaluates usage together with the requirement for a broad range of vehicles and price points, explicitly choosing **Uber** as the answer matching both criteria in the recorded evaluation.

**Capability illustrated:** The improvement in this comparison is in the synthesis of retrieved information. The selected RL response separates incompatible measures and ties its conclusion to the question’s constraints, while retaining the qualification that different definitions of usage can favor different candidates. These are observations about the recorded trajectories, not a claim about the current ride-hailing market.