arXiv:2602.01204v1 [cs.CL] 1 Feb 2026

ASTER：工具集成扩展推理的智能体扩展

xZhang [*1 2 3] 何泉 [*1] 郑振瑞 [4] 张宗章 [2 3] 何旭 [1] 李栋 [1]

摘要

强化学习（RL）已成为激发大语言模型（LLMs）长时程推理能力的主流范式。然而，通过强化学习扩展工具集成推理（TIR）仍面临挑战，原因在于交互崩溃：这是一种病态状态，模型无法维持多轮工具使用，反而退化为大量内部推理，仅进行琐碎的事后代码验证。我们系统地研究了三个问题：（i）冷启动监督微调如何形成智能体式的工具使用行为先验；（ii）冷启动轨迹的交互密度如何影响探索和下游强化学习结果；以及（iii）强化学习交互预算如何影响不同推理时预算下的学习动态和泛化能力。随后，我们提出**ASTER**（**A**智能体**S**扩展**T**工具**E**集成的**R**扩展推理），这是一种通过有针对性的冷启动策略、优先选用高交互密度轨迹来规避上述崩溃的框架。我们发现，仅含4K条高交互密度轨迹的小规模专家冷启动数据集能带来最强的下游性能，建立稳健的行为先验，从而在长时间强化学习训练中实现更优探索。大量评估表明，ASTER-4B在具有竞争力的数学基准测试中取得了最先进的成绩，在AIME 2025上达到90.0%，超越了包括DeepSeek-V3.2-Exp在内的领先开源模型。

1. 引言

大语言模型（LLM）通过强化学习（RL）在推理方面取得了显著进步，这有助于完善长链思维（思维链）（Wei 等人，2022年；Yao 等人，2023）。近期系统，如 OpenAI o1（Jaech 等人，2024年）和 DeepSeek R1（Guo 等人，2025年），展示了这种方法的有效性。通过采用错误纠正、问题分解和迭代改进等分析策略，这些模型在复杂推理任务中展现出强大的能力（Yeo 等人，2025年）。

尽管长思维链已取得成功，纯文本推理仍然存在固有的脆弱性，因为缺乏外部验证，微小错误可能不断累积，超出自我纠正机制的处理范围。工具集成推理通过为语言模型配备可执行工具来克服这一问题，利用外部反馈实现精确计算和中间验证，从而确保长时程推理任务的可靠性（Lin & Xu，2025年）。

现有的工具集成推理方法主要区别在于工具使用行为的初始化方式。零样本工具集成推理（Ze- roTIR）旨在通过强化学习直接从预训练基座模型中诱发工具使用，而不依赖有监督的冷启动数据。尽管概念上简单，ZeroTIR 仍面临两个根本局限。从优化角度来看，多轮工具集成推理训练往往不稳定。工具执行会引入非平稳反馈和向低概率词元的分布漂移，这些影响会在多轮过程中累积，往往导致梯度爆炸和收敛表现不佳（Wang etal.，2025年；Mai et al.，2025年；Xue et al.，2025年）。从能力角度来看，强化学习主要是强化已有能力，而非赋予全新的能力（Zhanget al.，2025年）。因此，ZeroTIR 仍受限于底层基础模型的表征和程序能力限制，并且始终难以胜任长时程规划、协同工具调用和程序级错误纠正。

另一种范式是在强化学习（Feng 等人，2025年；Shang 等人，2025年；Yu 等人，2025b）之前，先通过监督微调（SFT）在精心挑选、经工具增强的轨迹上对工具集成推理（TIR）进行初始化。这些轨迹通常通过以下方式获得：将纯文本思维链轨迹改写为代码交错形式（Feng 等人，2025年），或借助代码专用教师模型合成工具使用示范（Yu 等人，2025b）。尽管经验上取得了成功，现有的冷启动设计仍会通过轨迹构建引入强烈的归纳偏置。

[1]基础模型系，2012 Labs，华为 [2]新型软件技术国家重点实验室，中国南京大学 [3]人工智能学院，中国南京大学 [4]香港中文大学（深圳）。联系作者：何泉 <jiaqibbq@gmail.com>，张宗章 <zzzhang@nju.edu.cn>。

预印本。*2026*年*2*月*3*日。

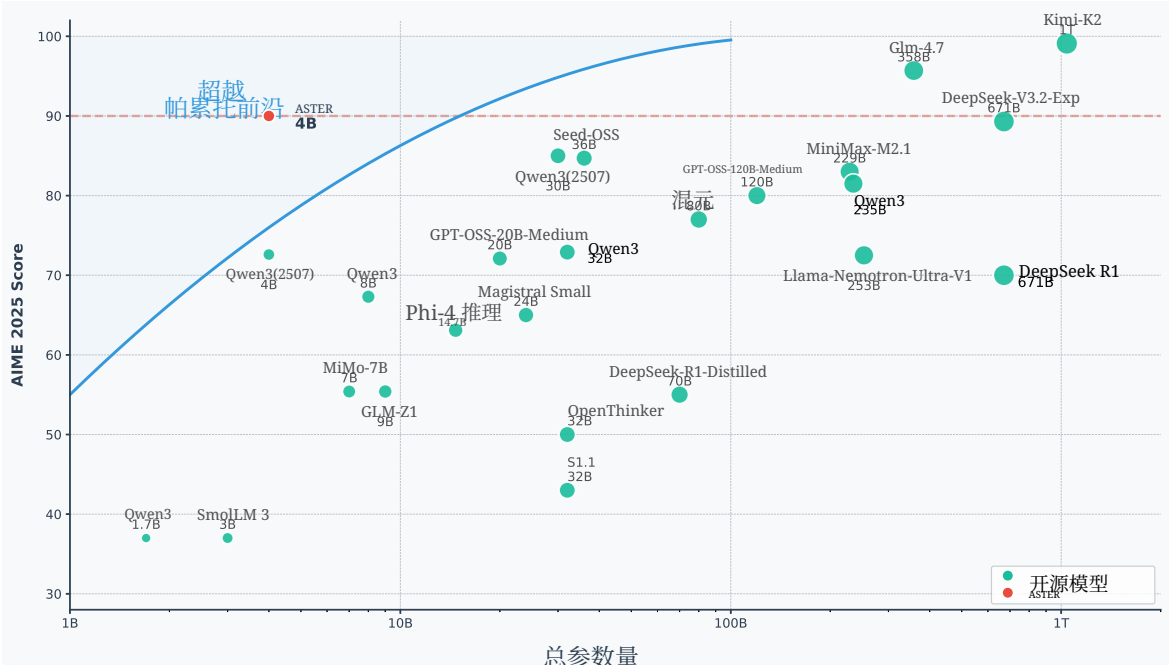


图 *1.* ASTER 展现出卓越的效率，在具有挑战性的 AIME 2025 基准测试中超越了规模大得多、实力更强的模型。它取得了 90.0 分，优于 DeepSeek-V3.2-exp（89.3/671B）。

ReTool（Feng 等人，2025年）基本保留了原有的纯文本推理结构，限制了真正由工具驱动的规划能力的形成。相比之下，代码专用教师模型生成的轨迹（Yu 等人，2025b）往往只包含简短、反应式的推理。这与慢思考模型所期望的审慎、系统2行为相悖。这些归纳偏置常常会导致后续强化学习训练期间出现交互崩溃：这是一种病态失效模式，模型无法持续进行多轮工具使用，反而退化为大量内部推理，最后仅进行简单的事后代码验证。

在本研究中，我们提出了 **ASTER** （**智**能体**扩**展与**工**具集成**扩**展**推**理），这是一个两阶段框架，将有针对性的冷启动监督微调阶段与强化学习相结合，以引导模型展现长时程、多轮工具使用行为。我们不以优化监督微调后的即时准确率为目标，而是研究冷启动监督微调如何形成多轮工具使用的行为先验。具体而言，我们提出以下问题：

- **RQ1（冷启动先验）**。冷启动监督微调在诱导智能体式工具使用行为先验方面发挥什么作用？不同的冷启动设计又如何塑造强化学习下游的工具使用策略与性能？

- **RQ2（交互密度）**。冷启动轨迹的交互密度如何影响探索行为以及后续的强化学习结果？

- **研究问题3（强化学习交互预算）**。强化学习交互预算如何影响学习动态和泛化，尤其是在推理时预算变化时？

通过广泛实验，我们发现交互密度是冷启动先验的关键属性。具体而言，我们使用 GPT-OSS-20B 合成工具增强轨迹，并从中整理出一个包含 4K 条样本、每条轨迹具有九轮以上工具交互轮次的小型专家数据集，将其用作冷启动监督微调先验。通过使用长时程、迭代式工具使用轨迹初始化模型，ASTER 形成了高熵行为先验，从而在后续强化学习过程中保留探索能力。与以推理为中心的初始化相比，这种设计会降低监督微调后准确率，但它能有效防止模型过早收敛到短时程、次优策略。因此，模型能够在强化学习过程中更充分地发挥能力扩展的优势，最终超越基于稀疏、短时程轨迹训练的基线方法。

基于这种交互密集的先验，ASTER 进一步采用多阶段强化学习流程，以稳定并优化扩展推理循环。实证结果表明，ASTER-4B 达到了最先进水平，在 30K 推理预算下，AIME 2025 的准确率达到 85.0%，HMMT 2025 达到 73.3%。尽管模型规模较小，它在一系列竞争激烈的数学基准测试中始终优于拥有 2350 亿参数的 Qwen3-235B-A22B-Thinking。此外，ASTER 展现出强大的测试时扩展能力：将推理预算增加至 90K 后，其在 AIME 2025 上的表现提升至 90.0%，这证实了交互密集型

训练能够释放可扩展的智能体智能。

2. 预备知识

监督微调 监督微调通过在高质量指令-响应对上进行下一个词元预测，使预训练大语言模型适应专门任务。在智能体推理领域，监督微调通常作为冷启动阶段，用于初始化多轮工具使用能力。构建此类训练数据的常见策略包括：将纯文本思维链解改写为代码交错格式（Feng 等人，2025年），或利用代码专用教师模型合成工具使用示范（Yu 等人，2025b）。这种初始化确立了后续强化学习所需的格式和基线交互能力。

强化学习 尽管监督微调确立了基础行为模式，但其效果受限于示范数据的质量和多样性。因此，后续采用强化学习来实现超越模仿的泛化，通过探索和反馈优化模型的推理能力。在数学推理和智能体任务中，强化学习将训练目标从最小化预测误差转变为最大化奖励信号——奖励信号通常依据最终答案的正确性或中间执行步骤的有效性来定义。组相对策略优化（GRPO）（Shao 等人，2024年）已成为一种重要算法。不同于依赖计算成本高昂的价值函数评论器的传统方法（如近端策略优化），GRPO 通过为单个查询 $\{o_1, o_2, ..., o_G\}$ 采样一组包含 G 个输出 q来优化策略。它利用组内的相对表现作为基线，从而显著降低内存开销并稳定训练。

优化目标最大化以下代理损失：

$$\mathcal{J}(\theta) = \mathbb{E}_{q\sim\mathcal{D},\,\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)}\left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\Big(r_{i,t}(\theta)\hat{A}_{i,t},\,\text{clip}\big(r_{i,t}(\theta),1-\epsilon,1+\epsilon\big)\hat{A}_{i,t}\Big)\right],$$

其中 $r_{i,\,t}(\theta) = \frac{\pi_\theta\,(o_{i,\,t}|q,\,o_{i,\,<t})}{\pi_{\theta_{\text{old}}}\,(o_{i,\,t}|q,\,o_{i,\,<t})}$ 是当前策略 π_θ 与旧策略 $\pi_{\theta_{\text{old}}}$ 之间的概率比（即用于在更新前进行采样的 ˆ 策略），而 $A_{i,\,t}$ 是根据组内相对奖励计算得到的优势。

3. 迈向最佳智能体扩展方案

在本节中，我们考察后训练流程，具体而言是冷启动监督微调设计与强化学习交互预-算，如何塑造大语言模型中智能体推理的涌现。我们研究冷启动监督微调是否会诱导一种行为先验，该先验支配模型在下游强化学习过程中如何规划、探索并与工具交互。我们围绕以下研究问题组织实证分析：

- **RQ1（冷启动监督微调和行为先验的作用）**。冷启动监督微调设计选择（例如轨迹模式）如何塑造诱导出的行为先验？该先验又如何影响后续强化学习中的工具使用策略和性能？
- **RQ2（监督微调交互密度的影响）**。冷启动监督微调的交互密度如何影响诱导出的行为先验以及后续强化学习中的性能？
- **RQ3（强化学习训练交互预算的影响）**。强化学习训练期间的交互预算如何影响学习动态和最终性能？当推理时工具预算发生变化时，它又如何影响测试时性能？

3.1. 实验设置

训练数据我们的监督微调数据集来自两个主要来源：(i) Skywork-OR 提供的 Skywork-OR1-RL-Data1[1] (He 等人，2025年），以及 (ii) OpenMath-Reasoning 从解题艺术（AoPS）论坛收集的 9.3 万道题目（Moshkov 等人，2025年）。我们仅保留真实最终答案为正整数的英文题目。随后，我们使用 GPT-OSS-20B-high 生成工具增强解答，并仅保留最终答案正确且工具执行成功率为 100% 的样本，得到包含 4.5 万条轨迹的初始数据集。对于强化学习训练，我们采用 DAPO-Math-17K[2], ，其中包含 1.7 万条与整数真实答案配对的提示。

基础模型我们使用 Qwen3-1.7B-Thinking 和 Qwen3-4B-Thinking-2507 作为基础模型。

冷启动监督微调 我们使用 LLaMA-Factory[3] 进行多轮指令微调。训练共进行 6 个周期，全局批次大小为 128，最大上下文长度为 32K 词元。我们采用恒定学习率： 3×10^{-5}，用于 Qwen3-1.7B/40亿。

强化学习 我们的框架继承VeRL[4]作为子模块，从而确保与上游更新兼容。我们使用组相对策略优化进行强化学习训练。遵循（Yu 等人，2025a），我们仅优化使用 Clip-Higher（$\epsilon_h = 0.28$）的令牌级策略梯度目标，并且

[1]Skywork-OR1 数据集[2] DAPO-Math-17K 数据集[3]GitHub: hiyouga/LlamaFactory[4]GitHub: volcengine/verl

模型	AIME2024 avg@16	AIME2025 avg@16	HMMT2025 avg@16	BeyondAIME 平均值@16
纯文本推理模型				
Qwen3-1.7B-Thinking	47.5	38.3	25.8	20.8
Qwen3-4B-Thinking-2507	76.7	72.7	48.1	43.6
OpenAI o3-mini（中等）	79.6	77.0	53.0	–
POLARIS-4B-Preview	81.2	79.4	58.7	–
OpenReasoning-Nemotron-7B	84.7	78.2	63.5	–
Qwen3-235B-A22B-Thinking	85.7	81.5	62.5	–
智能体推理模型				
ReTool-32B	72.5	54.3	–	–
rStar2-Agent-14B	80.6	69.8	52.7	–
DemyAgent-4B	72.6	70.0	52.9†	35.3†
ASTER-1.7B-SFT	19.4	19.0	11.3	6.4
ASTER-1.7B	64.6	59.6	47.5	26.3
ASTER-4B-SFT	62.5	54.6	43.3	27.4
ASTER-4B	82.3	85.0	73.3	53.9
使用 *90K* 推理预算	**85.8**	**90.0**	**77.1**	**61.7**

表 *1.* 在竞争性数学基准上的性能对比。我们报告 avg@16 的平均结果。所有评估指标均遵循 DeepSeek-R1 评估框架（温度 =0.6，top p $= 0.95$）。在官方性能数据可用时，我们优先采用这些数据；标有 † 的结果表示我们使用官方推荐配置进行的独立评估，**推理预算限制为 30K**。最佳分数以**粗体**标出，次佳分数加下划线。

不加入 KL散度惩罚。在数据收集方面，我们将 rollout提示批量大小设为 128，并为每个提示采样 $G = 8$ 条轨迹，以构建组基线。我们采用梯度累积，小批量大小为 64，从而在每个轨迹采样步骤中恰好进行两次参数更新。除非另有说明，步骤均指轨迹采样步骤。在所有实验中，我们使用统一的对话模板。为支持长时程工具使用和迭代式自我纠正，我们将每条轨迹允许的最大工具调用次数设为 50 次。

评估协议 我们在四个竞争性基准上评估数学推理能力：AIME2024、AIME2025、HMMT2025（二月）和 BeyondAIME（Seed 等人，2025年）。除非另有说明，所有评估均使用温度 0.6 和 top p $= 0.95$。所有其他评估参数均遵循相应基线论文推荐的设置。默认情况下，评估时使用 32K 词元的最大上下文长度。

基线方法 我们将我们的模型与多种最先进的基线方法进行比较，包括 OpenReasoning- Nemotron-7B（Ahmad等人，2025年）、Qwen3-235B-A22B- Thinking（Yang等人，2025年）、POLARIS-4B-Preview（An等人，2025年）、ReTool-32B（Feng等人，2025年），以及 rStar2-Agent-14B（Shang等人，2025年）和 DemyAgent- 4B（Yu等人，2025b年）。全部基线方法的性能数值均直接引自各自的论文。

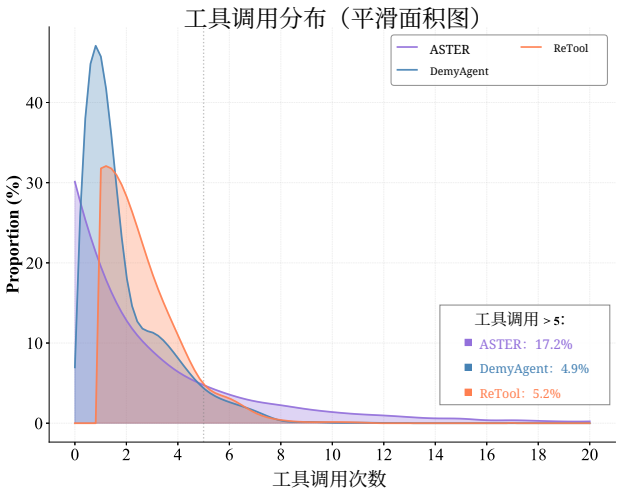
3.2. 冷启动监督微调与行为先验的作用（RQ1）

为回答 RQ1，我们以 Qwen3-4B-2507-Thinking 作为统一的基础模型进行受控比较，仅改变强化学习之前的冷启动初始化。我们考虑以下策略：

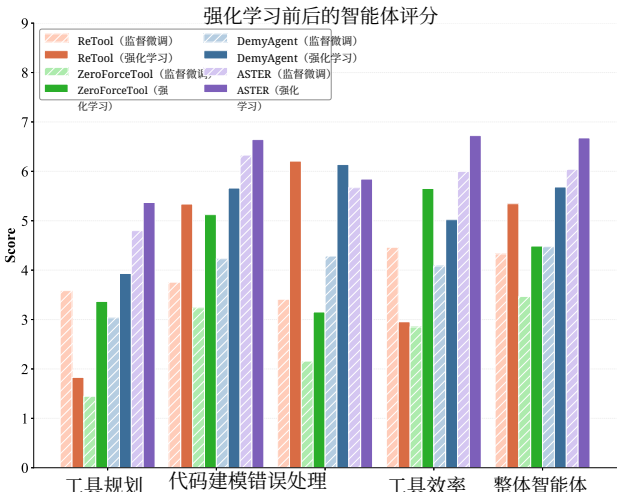
- **Zero**：不进行冷启动监督微调的直接强化学习训练。答对最终答案时奖励为 1，否则为 0。
- **ZeroForceTool**：不进行冷启动监督微调的直接强化学习训练，但采用工具约束奖励。只有最终答案正确且 观察到明确的工具使用时，奖励才为 1；否则为 0。
- **ReTool**：在 ReTool[5]上进行冷启动监督微调。ReTool 遵循“拼接式”合成范式，将轨迹中的计算步骤改写为代码片段。
- **DemyAgent**：在 DemyAgent[6]上进行冷启动监督微调，该数据集由 Qwen3-Coder-300亿-A3B 生成的端到端合成轨迹组成。
- **ASTER**：在我们精选的 45K 数据集上进行冷启动监督微调（见第 3.1节）。

为刻画冷启动数据集之间的差异，我们分析了它们的工具调用分布特性。图 2a 展示了工具调用次数的分布。ReTool 和 DemyAgent 的工具使用明显偏向稀疏：

[5]ReTool-SFT数据集[6] Open-AgentRL-SFT-3K数据集



(a) 不同冷启动数据集上的工具调用次数分布。



*(b)*智能体评判器比较。

图 *2.* 冷启动数据集的分布特性。(a) 工具调用次数分布表明，ReTool 和 DemyAgent 数据集严重偏向稀疏交互（1–2 次工具调用），而我们的数据集包含更多长时程轨迹。(b) 不同的冷启动策略会诱导出不同的行为先验，并使用智能体评判器评估其智能体能力。

大多数轨迹仅包含 1–2 次工具调用，超过 10 次调用的不到 2%（ReTool 的比例低至 0.1%）。相比之下，我们的监督微调数据集大幅提高了长时程且密集使用工具的轨迹占比。为排除数据集难度这一混杂因素，我们使用 Qwen3-40亿-2507-Thinking 的 8 个样本测量每道题的准确率；图 7a显示题目难度相近，这表明关键差异在于交互模式，而非问题难度。

为了评估初始化如何在冷启动监督微调和随后的强化学习之后塑造内在智能体行为，我们使用基于 GPT-5 的智能体评判器，并采用固定的评分标准和提示模板（第 A.4节）进行评分，维度包括规划、代码建模、错误处理、工具效率以及整体智能体能力。如图 2b所示，所有冷启动变体在各维度上的得分均高于 **Zero** ，且 **ASTER**取得了最强且最一致的提升，表明不同的冷启动监督微调初始化会诱导出不同的 行为先验。

图 3 总结了强化学习训练在任务性能和工具调用次数方面的动态。**ASTER** 呈现出一种独特模式：在整个训练过程中持续进行密集工具使用，同时始终优于基线方法。

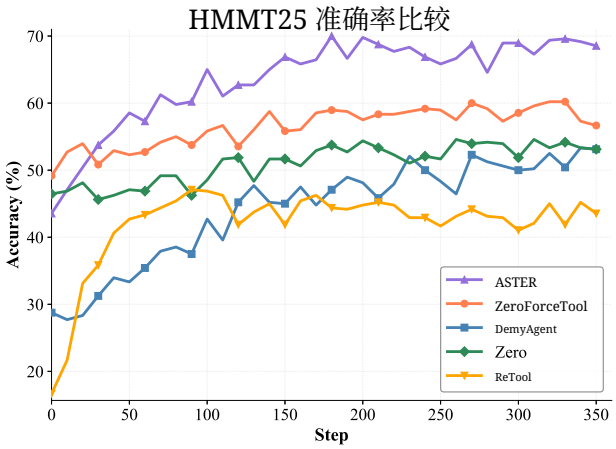
相比之下，基线方法表现出明显的局限性。**Zero** 持续倾向于最少的工具使用。**ZeroForce-Tool** 起初会增加工具使用，但往往退化为浅层验证行为，未能习得真正的多步交互，并出现交互崩溃。**ReTool** 同样在最初上升后降至较低的工具使用水平，并迅速停滞在次优性能。最后，尽管**DemyAgent** 达到了中等程度的工具使用（≈ 4 次调用），但其短视野训练轨迹限制了复杂任务所需的长交互链学习，从而制约了最终准确率。总体而言，冷启动并非普遍有益：当演示以稀疏、短视野的工具交互为主（如 ReTool 和 DemyAgent）时，所诱导的先验会使学习偏向浅层验证行为，并限制后续强化学习的收益，从而无法超越**Zero**。

要点 1：冷启动监督微调决定强化学习的行为先验

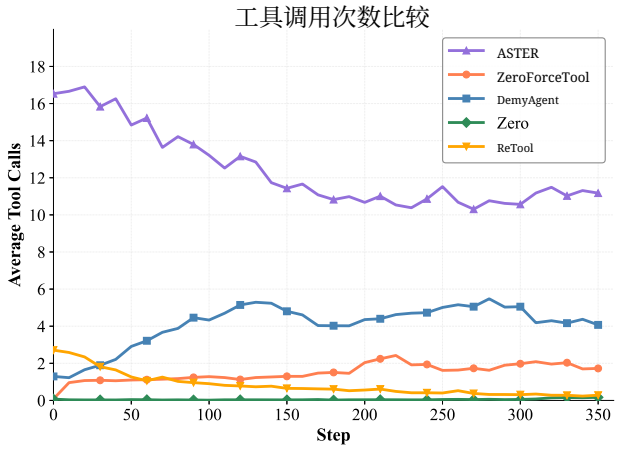
冷启动监督微调的效果并非一概而论；关键取决于示范中编码的交互模式。稀疏且短时程的轨迹会形成一种偏向浅层、验证式工具使用的行为先验，从而限制强化学习下的策略改进。相比之下，只需纳入少量长时程、高交互密度的轨迹（ASTER），就足以引导学习转向持续的多轮工具使用，并提升下游性能。

3.3. 监督微调的交互密度的影响（RQ2）

为研究冷启动监督微调中交互密度的影响，我们按照每条轨迹中的工具调用次数对第 3.1 节所述的 45K 数据集进行分层。具体而言，我们构建了三个子集：至多包含 1 次工具调用（22K 条轨迹）、至多包含 5 次工具调用（37K 条轨迹）以及至少包含 9 次工具调用（4K 条轨迹）。我们分别使用每个子集对模型进行冷启动，然后运行相同的强化学习流程。在这项研究中，强化学习训练和评估使用的上下文长度均为 18K。



(a) HMMT25 准确率比较。



*(b)*工具调用次数比较。

图 *3*。 RQ1 分析：Col 起始策略塑造下游强化学习动态。(a) 和 (b) 展示了合成教师的影响 r 模式，展示了性能 差距和不同的工具使用策略。

图 4 展示了强化学习期间 AIME25 的准确率；第 A.3节报告了相应的训练熵。我们观察到最终准确率呈现出一致的排序：从高交互密度轨迹（≥ 9 次工具调用）初始化的模型表现最佳，其次是完整数据集，然后是 ≤ 5 子集，而 ≤ 1 子集表现最差。除了最终准确率之外，≥ 9初始化还使训练过程中的熵始终保持较高水平，这表明模型进行了更广泛的探索，而非过早坍塌。综合来看，这些结果表明，注重工具使用的冷启动监督微调会使后续强化学习动态朝着更持久地使用工具的方向发展，从而帮助策略发现质量更高的解。

要点（研究问题2）：交互密度至关重要

使用高交互密度轨迹进行冷启动监督微调，可建立更强的工具使用行为先验，从而在强化学习期间持续保持较高的训练熵，进而获得更优的最终任务性能。

3.4. 强化学习训练中交互预算的影响（研究问题3）

研究问题3考察强化学习交互预算如何影响涌现的推理行为和测试时扩展，并比较在训练时工具预算受限与充裕的情况下学到的策略。

我们使用第 3.3 节介绍的 4K 高交互轨迹，对 Qwen3-4B-2507-Thinking 进行冷启动。在强化学习训练期间，我们仅调整每条轨迹的最大工具调用次数，将预算设为 10 次或 50 次调用，同时保持其他所有超参数与主要配置一致。所有强化学习训练均采用 18K 的上下文长度。

评估时，模型将在扩展后的 32K

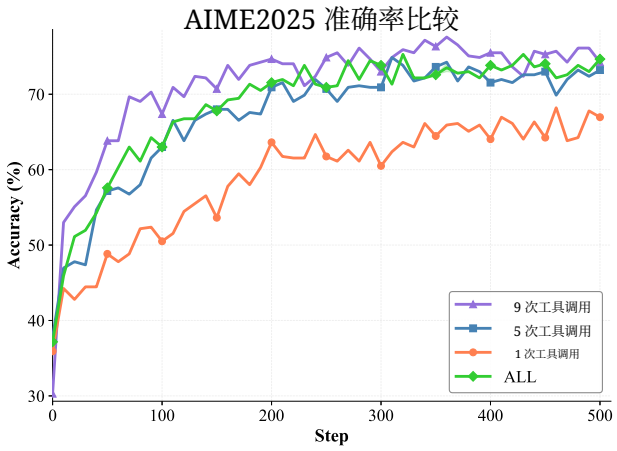


图 *4.* 工具密集型监督微调在 AIME25 上取得了更高的性能。

上下文窗口中进行测试。我们系统地将推理时工具预算从每条轨迹 2 次调用调整到 256 次调用，从而能够刻画训练时交互约束如何影响不同部署场景下的性能扩展。

图 5 显示出明显的预算对齐效应。扩展曲线在中等推理预算处相交，超过该点后，相对优势便会逆转。当推理交互资源充足时，经过高交互容量训练的模型峰值性能显著更高，在 256 次工具调用时准确率达到 83.0%，而受限训练模型为 77.7%。相反，在推理预算紧张时，经过低交互容量训练的模型明显更有效，仅用 2 次工具调用便达到 51.5% 的准确率，而经过高预算训练的模型为 32.3%。

总的来说，这些结果凸显了在

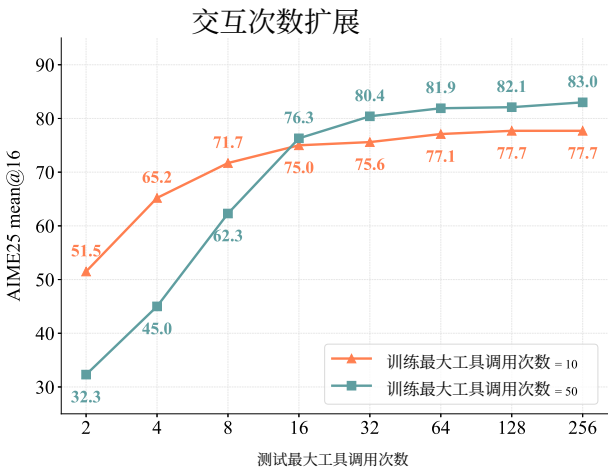


图*5*。 在不同交互约束下训练的模型，其测试时性能随推理时工具预算变化的扩展情况。在较大的推理预算下，使用更高交互预算（50次调用）训练的模型占据优势；而当推理预算受到严格限制时，交互受限训练的模型表现更好。

用于工具增强推理的仅基于结果的强化学习中，对齐训练时与部署时交互预算的重要性。训练与推理约束条件不匹配会显著降低性能，而预算对齐则能使模型更接近其最佳运行状态。

要点（RQ3）：训练时交互预算至关重要

随着推理时工具预算增加，较大的训练时交互预算能够带来更好的测试时扩展表现。

4. ASTER方案

基于RQ1至RQ3的洞见，我们提出了ASTER的一种具体实现。

遵循RQ1和RQ2，我们在4K精选子集上进行训练，并使用超过九个交互轮次。遵循RQ3，我们将每条训练轨迹中的工具调用次数上限设为50次。

为进一步提升训练的稳定性和效率，我们采用了最大上下文长度逐步增加的两阶段强化学习课程：阶段 1 为 18K 词元，阶段 2 为 32K 词元。在过渡到阶段 2 之前，我们会筛除那些在最后一个训练轮次的所有采样轨迹中都持续得出正确结果的提示，从而将后续训练集中在更具挑战性的实例上。

为评估 ASTER 的有效性，我们在表1中进行了比较分析。结果表明，ASTER 在所有评估基准测试中均实现了最先进的性能，显著优于纯文本推理模型和现有的智能体推理系统。

值得注意的是，在 9 万的推理预算下，ASTER-4B 在 AIME2024、AIME2025、HMMT2025 和 BeyondAIME 上分别取得了 85.8%、90.0%、77.1% 和 61.7% 的成绩。与现有智能体推理模型相比，ASTER 的优势更加显著：ASTER-4B 在 AIME2024、AIME2025 和 HMMT2025 上的表现分别比最佳智能体基线模型（rStar2-Agent-14B）高出 5.2% 、 20.2% 和 24.2%。值得一提的是，尽管 ASTER-4B 基于一个 40 亿参数模型训练而成，其表现仍可媲美甚至超越 Qwen3-235B-A22B-Thinking 和 OpenAI o3-mini 等规模大得多的模型，同时保持着更出色的工具集成推理能力。

我们在图 6中展示了涵盖两种模型规模（17亿、40亿）的全面实验结果，以刻画 ASTER 框架的训练动态和行为模式。我们的分析聚焦于训练过程中显现的三个关键实证观察。

冷启动性能恢复 经过冷启动监督微调后，我们发现模型在数学推理基准测试上的性能显著下降，准确率大幅低于监督微调前基础模型的基线性能。然而，随着强化学习训练的推进，模型准确率逐渐恢复，达到并随后超过监督微调前的基线水平。

强化学习训练动态 如图 6所示，在强化学习训练的第一阶段，我们观察到每条轨迹的平均工具调用次数系统性地减少，并在若干训练轮次后稳定在较低水平。这一减少反映出模型逐渐适应更高效的工具使用模式，学会在保持解答质量的同时消除冗余工具调用。过渡到上下文长度扩展至 32K 词元的第二训练阶段后，我们记录到多项行为指标同步改善：平均响应长度增加，工具调用频率上升，基准测试准确率持续提升。这一模式表明，扩展后的上下文容量使模型能够处理更具挑战性的问题，这些问题需要更长的推理轨迹和更频繁的工具交互；与此同时，工具使用量在初期的减少为高效利用工具奠定了基础，避免了不必要的调用。

模型规模效率差异 在不同模型规模下，我们观察到工具使用效率存在系统性差异。具体而言，较小模型（17亿）每条轨迹的工具调用次数高于中等规模模型（40亿）。平均响应长度也呈现相同趋势：较小模型生成的轨迹更长，

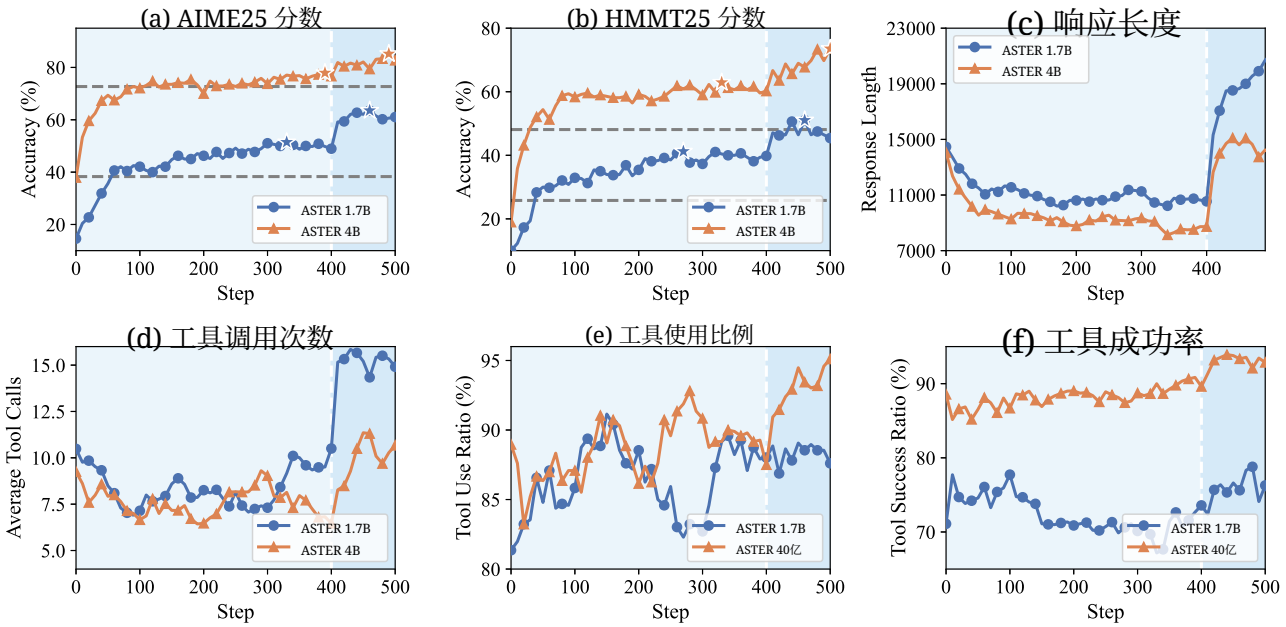


图 *6.* 随着训练的推进，ASTER逐渐恢复到监督微调前的性能水平，并显著超越该水平。模型的工具调用频率和响应长度先下降后上升，而工具调用率和执行成功率则逐步提升。

而较大模型则以更简洁的推理路径和更少的工具调用，实现相当或更优的性能。

5. 相关工作

工具集成推理（TIR）使大语言模型能够将自然语言推理与外部工具执行（例如代码解释器或搜索引擎）交替进行，从而提高其在需要精确符号运算或流程操作的任务上的可靠性。

以往工作通常通过精选的工具使用轨迹进行监督学习来训练工具集成推理，例如 ToRA（Gou 等人，2024年）和 QwenMath-TIR（Yang 等人，2024年）；然而，这些监督微调流程通常依赖高质量标注和/或精心设计的格式约束，这可能会限制其在不同工具和领域中的可扩展性与稳健性。

其他研究分析了训练动态和基于强化学习的工具使用：SimpleTIR（Xue 等人，2025年）筛除弱信号轨迹，ZeroTIR（Mai 等人，2025年）探究了涌现式工具调用的缩放定律，而 ARTIST（Singh 等人，2025年）则通过基于结果的强化学习，在没有步骤级监督的情况下学习多轮工具决策。

冷启动合成（Feng 等人，2025年）也已得到探索：ReTool 将纯文本解改写为工具集成轨迹，但可能破坏工具调用的因果逻辑；而教师生成轨迹（Yu 等人，2025b）可能因教师效率较高而工具使用稀疏。

相比之下，ASTER 通过一小组轨迹来启动仅基于结果的强化学习这些高交互性轨迹包含因果规划—执行—解读—改进循环，并在有状态的Python沙箱中进行显式状态跟踪，从而在多阶段组相对策略优化训练期间促进探索，并减少训练坍缩为短视界验证的情况。

6. 结论

在这项工作中，我们提出 ASTER，以缓解工具集成推理中的交互崩溃。我们表明，冷启动监督微调所形成的行为先验在很大程度上决定了强化学习的扩展效果；强调交互密度——优先考虑长时程轨迹，而非监督微调后的即时准确率——能够保留探索，并避免过早收敛到浅层策略。

实证结果表明，ASTER-4B 达到了最先进的性能，在 AIME 2024 上取得 85.8% 的成绩，在 HMMT 2025 上取得 77.1% 的成绩。值得注意的是，它的表现显著优于规模大得多的系统，包括 Qwen3-235B-A22B-Thinking 和 rStar2-Agent-14B，这证明只要具备更强的工具集成能力，小型模型也能媲美庞大的基线方法。

我们的分析表明，交互密集的初始化能够维持更高的训练熵，并与最终的强化学习性能相关，这凸显了智能体扩展的一项关键原则：冷启动应优先考虑行为多样性与探索，而非短期指标，从而在参数扩展之外实现可扩展的多步问题求解。

影响声明

本文研究如何通过强化学习扩展工具集成推理，并提出ASTER以缓解交互崩溃，从而实现更可靠的多步工具使用与长时程问题求解。若负责任地采用，这些能力有望提升复杂推理工作流（例如数学问题求解与研究辅助）的效率与可及性，并可能降低对超大模型的依赖。与此同时，更强的工具使用型智能体可能放大滥用风险（例如自动执行有害任务），并在与外部工具或代码执行环境交互时引入安全问题。因此，我们强调部署应使用沙箱工具、最小权限访问、监控与人工监督，并鼓励未来针对工具集成的强化学习智能体开展稳健的安全评估研究。

参考文献

Ahmad, W. U., Majumdar, S., Ficek, A., Narenthiran, S., Samadi, M., Huang, J., Jain, S., Noroozi, V., and Ginsburg, B. Opencodereasoning-ii：一种通过自我批评进行测试时扩展的简单方法。*arXiv* 预印本 *arXiv:2507.09075*，2025年。

An, C., Xie, Z., Li, X., Li, L., Zhang, J., Gong, S., Zhong, M., Xu, J., Qiu, X., Wang, M., 等。Polaris：面向高级推理模型扩展强化学习的一种后训练方案，2025年。网址 *https://hkunlp.github.io/blog/2025/Polaris*，2025年。

Feng, J., Huang, S., Qu, X., Zhang, G., Qin, Y., Zhong, B., Jiang, C., Chi, J., and Zhong, W. ReTool：面向大语言模型战略性工具使用的强化学习。*arXiv* 预印本 *arXiv:2504.11536*，2025年。

Gou, Z., Shao, Z., Gong, Y., Yang, Y., Huang, M., Duan, N., Chen, W., 等。Tora：一种用于数学问题求解的工具集成推理智能体。发表于国际学习表征会议（*ICLR*），2024年。

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., 等。Deepseek-r1：通过强化学习激励大语言模型的推理能力。*arXiv* 预印本 *arXiv:2501.12948*，2025 年。He, J., Liu, J., Liu, C. Y., Yan, R., Wang, C., Cheng, P., Zhang, X., Zhang, F., Xu, J., Shen, W., 等。Skywork 开放推理器 1 技术报告。*arXiv* 预印本 *arXiv:2505.22312*，2025年。Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., 等。OpenAI o1 系统卡。*arXiv* 预印本 *arXiv:2412.16720*，2024年。

Lin, H. 和 Xu, Z. 理解工具集成推理。*arXiv* 预印本 *arXiv:2508.19201*，2025年。

Mai, X., Xu, H., Li, Z.-Z., W, X., Wang, W., Hu, J., Zhang, Y. 和 Zhang, W.。智能体强化学习扩展定律：通过自发代码执行进行数学问题求解的智能体强化学习。发表于神经信息处理系统进展（*NeurIPS*），2025年。

Moshkov, I., Hanley, D., Sorokin, I., Toshniwal, S., Henkel, C., Schifferer, B., Du, W. 和 Gitman, I.。Aimo-2 获胜方案：利用 openmathreasoning数据集构建最先进的数学推理模型。*arXiv* 预印本 *arXiv:2504.16891*，2025年。

Seed, B., Chen, J., Fan, T., Liu, X., Liu, L., Lin, Z., Wang, M., Wang, C., Wei, X., Xu, W., 等。Seed1.5-thinking：通过强化学习推进卓越推理模型的发展。*arXiv* 预印本 *arXiv:2504.13914*，2025年。

Shang, N., Liu, Y., Zhu, Y., Zhang, L. L., Xu, W., Guan, X., Zhang, B., Dong, B., Zhou, X., Zhang, B., Xin, Y., Miao, Z., Li, S., Yang, F. 和 Yang, M. rstar2-智能体：智能体推理技术报告。*ArXiv* 预印本 *arXiv:2508.20722*，2025年。

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y. 和 Guo, D. Deepseek-math：突破开放语言模型数学推理的极限。*arXiv* 预印本 *arXiv:2402.03300*，2024年。

Singh, J., Magazine, R., Pandya, Y., 和 Nambi, A. 通过强化学习实现大语言模型的智能体推理与工具集成。*arXiv* 预印本 *arXiv:2505.01441*，2025年。

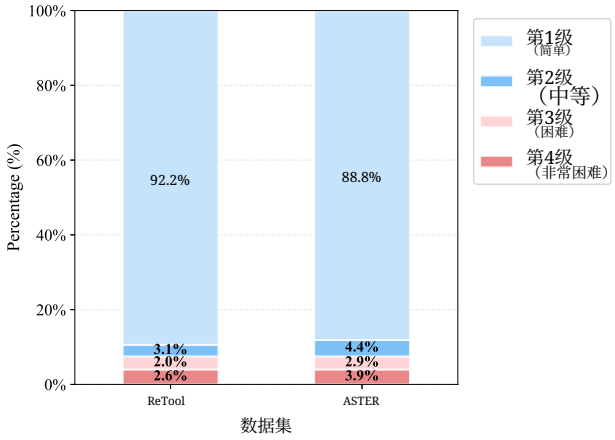
Wang, Z., Wang, K., Wang, Q., Zhang, P., Li, L., Yang, Z., Jin, X., Yu, K., Nguyen, M. N., Liu, L. 等。Ragen：通过多轮强化学习理解大语言模型智能体的自我进化。*arXiv* 预印本 *arXiv:2504.20073*，2025年。

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V. 和 Zhou, D. 思维链提示能够引发大语言模型的推理。载于神经信息处理系统进展（*NeurIPS*），2022年。

Xue, Z.、Zheng, L.、Liu, Q.、Li, Y.、Zheng, X.、Ma, Z. 和 An, B. SimpleTIR：用于多轮工具集成推理的端到端强化学习。载于国际学习表征会议（*ICLR*），2025年。

Yang, A., Zhang, B., Hui, B., Gao, B., Yu, B., Li, C., Liu, D., Tu, J., Zhou, J., Lin, J. 等。Qwen2.5-math 技术报告：通过自我改进迈向数学专家模型。*arXiv* 预印本 *arXiv:2409.12122*，2024年。

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C. 等。Qwen3 技术报告。*arXiv* 预印本 *arXiv:2505.09388*，2025年。Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y. 与 Narasimhan, K.。思维树：利用大语言模型进行审慎问题求解。收录于神经信息处理系统进展（*NeurIPS*），2023年。Yeo, E., Tong, Y., Niu, M., Neubig, G. 与 Yue, X.。揭秘大语言模型中的长思维链推理。*arXiv* 预印本 *arXiv:2502.03373*，2025年。Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L. 等。DAPO：大规模开源大语言模型强化学习系统。*arXiv* 预印本 *arXiv:2503.14476*，2025年a。Yu, Z., Yang, L., Zou, J., Yan, S. 与 Wang, M.。揭秘智能体推理中的强化学习。*arXiv* 预印本 *arXiv:2510.11701*，2025年b。Zhang, C., Neubig, G. 与 Yue, X.。论预训练、中期训练与强化学习对推理语言模型的相互作用。*arXiv* 预印本 *arXiv:2512.07783*，2025年。



（*a*）ReTool 和 ASTER 数据集的问题难度分布。难度按每道题的准确率定义：简单（≥ 0.75）、中等（$[0.50, 0.75)$）、困难（$[0.25, 0.50)$）和非常困难（< 0.25）。

组	Name	值
Data	训练批次大小	128
	最大提示长度	2048
	最大响应长度	16384/30720
执行者	lr	1e-6
	PPO小批次大小	64
	使用_KL_损失	否
	裁剪比率下限	0.20
	裁剪比率上限	0.28
	熵系数	0
轨迹采样	n (每个提示的组大小)	8
	助手最大轮数	50

(b) 与 verl 配合使用的超参数。

A. 实验细节

A.1. 数据集难度分布和强化学习训练设置

为刻画每个数据集的难度分布，我们使用 Qwen3-4B-2507-Thinking 作为基础模型，并针对每道题采样 8 个响应。我们将 8 个响应中正确样本的比例计算为每道题的准确率。随后，以 0.25 为步长，根据准确率阈值将题目划分为四个难度等级。如图 7a所示，简单题（准确率在 $[0.75, 1.0]$内）的比例在两个数据集中分别为 92.2% 和 88.8%，这表明在相同解码配置下，对于 40亿模型而言，两者的难度相当。

我们使用`verl`库实施强化学习（RL）训练。主要超参数汇总于图 7b中。在主要实验中，我们采用两阶段训练计划，将最大响应长度 设置为阶段 1 的 16,384 和阶段 2 的 30,720。我们禁用 KL 正则化，以避免过度限制长时域探索，尤其是在强化学习的早期阶段。

在 RQ1 中，对于不同教师模式冷启动策略的消融实验，我们将训练和评估的上下文长度均固定为 32K，并保持所有其他超参数不变。

在 RQ2 中，对于交互密度的消融实验，我们将强化学习训练和评估的上下文长度均设为 18K。

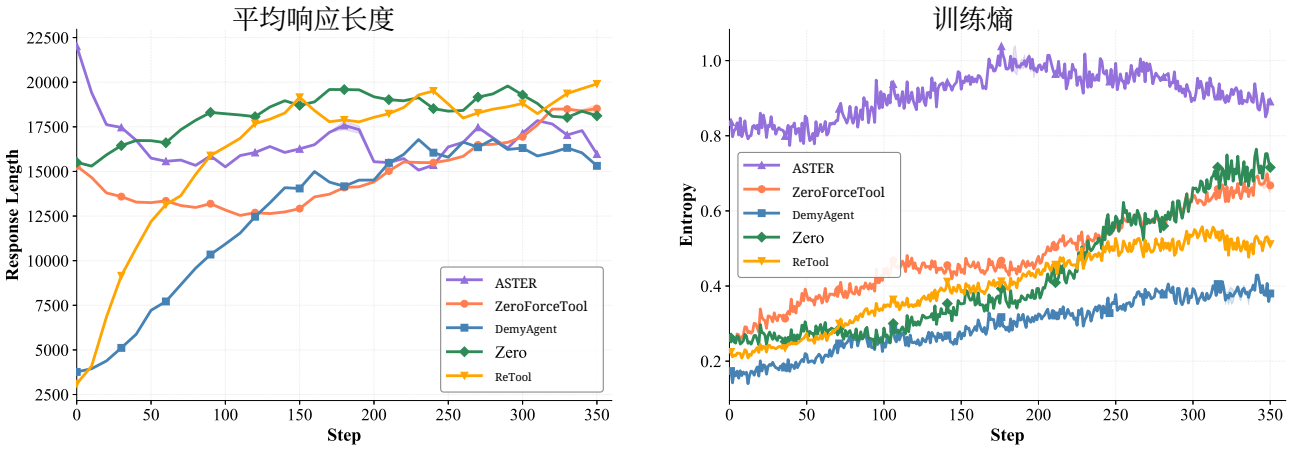
在研究问题3中，模型在 18K 的上下文长度下进行训练，同时将工具调用的最大次数分别限制为 10 次和 50 次。评估期间，我们将上下文长度扩展至 32K，以评估泛化能力。

A.2. RQ1补充实验结果

为了进一步了解冷启动策略如何塑造行为先验并影响后续强化学习训练，我们分析了其他训练时行为指标，包括响应长度和训练熵。

图 8 展示了采用不同教师模式冷启动策略（Zero、ZeroForceTool、ReTool、DemyAgent 和 ASTER）初始化的模型的训练动态，揭示了它们在响应长度和训练熵方面呈现出的不同轨迹。

训练熵为理解探索提供了一个有用视角：它反映了模型在训练期间生成过程的不确定性，并可作为探索能力的替代指标。熵越高，表明模型在决策空间中的探索范围越广；而熵值迅速下降，则意味着模型可能过早收敛到狭窄的行为模式。如图 8b所示，**ASTER**在整个训练过程中始终保持最高的训练熵，这表明我们的高交互密度冷启动策略形成了一种鼓励广泛探索的行为先验，而不是让模型过早收敛到过于短视的仅验证行为。这种更强的探索能力有助于模型在强化学习期间更好地搜索解空间，从而提升最终性能。相比之下，基线策略（尤其是 **Zero** 和 **ReTool**）的熵值较低且下降得更快，这与限制探索并导致模型更早收敛到次优



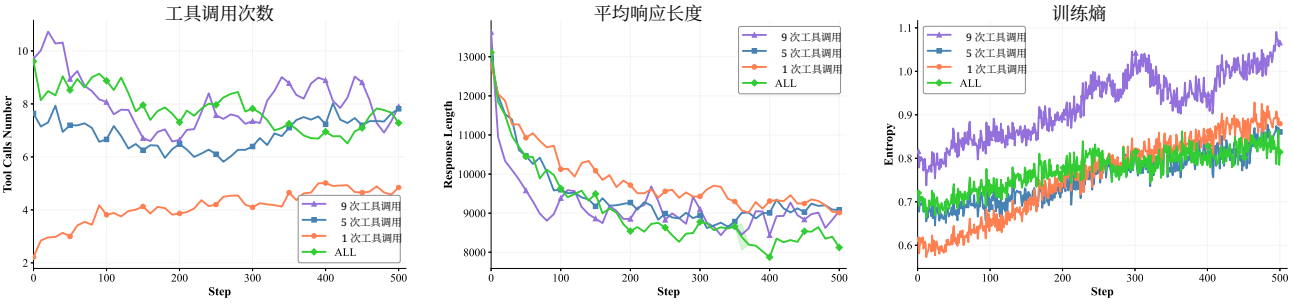
(a) 不同教师模式下的响应长度演变。 *(b)* 不同教师模式下的训练熵演变。

图*8.* 训练动态 在不同教师模式冷启动策略下。(a) 验证集上的响应长度。(b) 训练 g
熵在训练过程中的变化，反映 探索—利用动态。

模式。

A.3. RQ2的补充实验结果

为进一步验证交互密度对行为先验的影响，我们分析了按工具调用次数分层的数据冷启动模型的训练动态。图 9 展示了不同初始化密度下工具调用次数、响应长度和训练熵的变化。我们发现，交互密度较高的冷启动会带来更高的训练熵，表明其探索能力更强。此外，训练期间的工具使用模式与初始冷启动分布高度一致，这表明交互密度能够有效塑造行为先验。值得注意的是，初始化时交互密度较高的模型在训练早期会迅速缩短响应长度，这表明工具使用能够简化推理并提高简洁性。



(a) 训练过程中的工具调用次数。 *(b)*训练过程中的响应长度。 *(c)* 训练过程中的训练熵。

图 *9*。 不同交互密度下冷启动的训练动态。从左到右依次为：训练过程中的工具调用次数、响应长度和训练熵。

A.4. 智能体评判器

我们使用基于 GPT-5 的智能体评判器，配合固定的评分标准和以下提示模板，对规划、代码建模、错误处理、工具效率以及整体智能体能力进行评分。

智能体评判器提示模板

你是一名严格的智能体数学问题求解评估者。你将收到一位
人工智能智能体的完整轨迹，包括推理、工具调用、代码和中间输出。你的任务是评估该智能体的智能体
能力，而非最终答案的正确性。忽略轻微的语法错误和格式问题。
针对以下每个维度：
- 评定一个 1 到 10 的整数分数
- 评分仅基于轨迹中的可观察行为
- 将评分锚点作为严格参考，而非模糊指南

1. 工具规划（评分 1–10）
问题：智能体是否在采取行动之前规划了其工具使用？
评分锚点：
1–2：没有明显的计划。工具使用（如有）是被动应对或偶然发生的。
3–4：目标意识薄弱。只有在失败后才开始规划。
5–6：有初步计划，但缺少关键步骤或依赖关系。
7–8：有明确的多步骤计划。会预先考虑工具使用并安排好顺序。
9–10：具备专家级的预见能力。计划包含应急方案、检查和验证步骤。

2. 计算建模（评分 1–10）
问题：智能体是否正确地将数学结构转化为代码？
评分锚点：
1–2：把代码当作计算器使用。没有抽象或结构。
3–4：使用临时循环或条件语句。与数学结构的契合度较差。
5–6：正确运用暴力计算、模拟或函数分解。
7–8：采用恰当的计算范式（动态规划、搜索、不变量）。
9–10：模型与数学结构高度契合。高效、严谨且可扩展。

3. 错误诊断与恢复（评分 1–10）
问题：智能体在多大程度上能够发现并修正自身错误？
评分锚点：
1–2：重复犯错，忽略失败信号。
3–4：仅修复语法问题，逻辑错误仍然存在。
5–6：注意到输出不正确，并尝试修复。
7–8：找出根本原因并检验假设。
9–10：通过验证和最小复现案例进行系统调试。

4. 工具使用效率（评分 1–10）
问题：智能体是否在最适当的时候使用了工具？
评分锚点：
1–2：进行冗长的手动推理，而不是使用工具。
3–4：偶尔使用工具；错过明显的使用机会。
5–6：推理与工具使用之间的平衡较为合理。
7–8：积极且及时地使用工具。
9–10：对于所有非简单操作，工具选择都接近最优。

5. 整体智能体质量（评分 1–10）
问题：该智能体作为工具使用型问题求解器的能力有多强？
指导：
- 这并非之前各项评分的平均值。
- 请考虑规划、建模、调试和工具使用之间的协调情况。
评分锚点：
1–3：行为零散，规划和执行能力较弱。
4–6：能力表现参差不齐，存在明显的结构性弱点。
7–8：智能体行为强劲且连贯。
9–10：具备专家级自主性，且推理与工具协同出色。

B. 输出示例

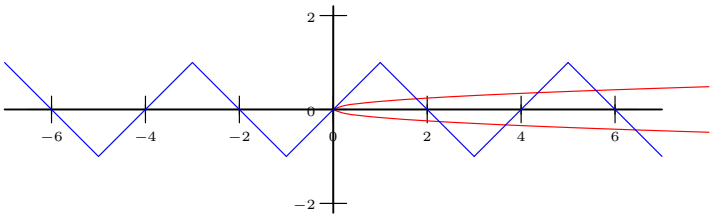
案例研究中的 AIME25 题目

示例案例研究：AIME题目

问题：
定义一个分段线性函数如下：

$$f(x) = \begin{cases} x & \text{if } -1 \le x < 1 \\ 2 - x & \text{if } 1 \le x < 3 \end{cases}$$

并且 $f(x+4) = f(x)$ 对所有实数 x均成立。 $f(x)$ 的图像呈现下方所示的锯齿形图案。抛物线 $x = 34y^2$ 与 $f(x)$ 的图像相交于有限个点。所有这些交点的 y 坐标之和可表示为 $\frac{a+b\sqrt{c}}{d}$，其中 a、 b、 c和 d 均为正整数，且 a、 b、 d 的最大公约数为 1， c 不能被任何素数的平方整除。求 $a+b+c+d$。图像



标注答案：
$\boxed{259}$

3

图 *10.* AIME2025-I 第 11 题

ASTER-4B 在 AIME 2025 中的轨迹。这是 48轮交互的第一部分。

ASTER-4B 的模型输出（第 1 轮） [初步分析]

（分别处理每条周期线段，并在每段上求解相应的二次方程）.....我们需要分别处理每条线段，$n \geq 0$。但还要考虑到，当 n 较大时，约束条件可能不成立；不过我们需要找到约束条件不成立之前的情况。但我们还需要考虑 n 为负数的情形吗？我们已经确定了 $n \geq 0$。不过，为了确认一下，来测试线段 A 中 n 为负数的情况，例如 n=-1 等。没有实数解。因此只有 $n \geq 0$。现在，我们可以从 0 开始暴力枚举 n，直到比如 10，计算解并检查约束条件。我们用 Python 来做。

```
import sympy as sp
n = sp.symbols('n', integer=True, nonnegative=True)
solutions = []
# Let's compute for n from 0 to maybe 10
for n_val in range(0, 10):
    # Segment A
    y1 = (1 + sp.sqrt(1 + 544*n_val)) / 68
    y2 = (1 - sp.sqrt(1 + 544*n_val)) / 68
    for y in [y1, y2]:
        x_val = 34 * y**2
        # check interval [4n-1, 4n+1)
        okA = (sp.S(4*n_val - 1) <= x_val) and (x_val < sp.S(4*n_val + 1))
        if okA:
            solutions.append((n_val, 'A', sp.simplify(y), sp.simplify(x_val)))
    # Segment B
    D = 544*n_val + 273  # numeric for each n_val
    y3 = (-1 + sp.sqrt(D)) / 68
    y4 = (-1 - sp.sqrt(D)) / 68
    for y in [y3, y4]:
        x_val = 34 * y**2
        okB = (sp.S(4*n_val + 1) <= x_val) and (x_val < sp.S(4*n_val + 3))
        if okB:
            solutions.append((n_val, 'B', sp.simplify(y), sp.simplify(x_val)))
solutions[:10]
```

执行结果：(0, 'A', 1/34, 1/34), (0, 'A', 0, 0), ...

ASTER-4B 的模型输出（第 8 轮） [确定有限范围]

模型推测，虽然抛物线无限延伸，但交点只会出现在有限的周期范围内 n。它编写代码，逐步遍历 n，直到找不到解为止。

```
max_n=0
for n_val in range(0,100):
found=False
# Check Segment A and B for existence of valid solutions
# ... (code calculating y1, y2, y3, y4 and checking intervals) ...
if found:
    max_n=n_val
max_n
```

执行结果：8

4

ASTER-4B 在 AIME 2025 中的轨迹。这是 48轮交互的第二部分。

ASTER-4B 的模型输出（第14轮） [符号求和]

确定交点仅在 $n \in [0, 8]$ 时出现后，模型继续计算所有 y-坐标。

```
sols=[]
for n_val in range(0,9):
# Segment A calculations
y1 = (1 + sp.sqrt(1+544*n_val))/68
y2 = (1 - sp.sqrt(1+544*n_val))/68
for y in [y1,y2]:
# ... check constraints ...
    sols.append(sp.simplify(y))
# Segment B calculations
# ... check constraints ...
sols
```

执行结果： [1/34, 0, -1/68 + sqrt(273)/68, ..., -1/68 + 5*sqrt(185)/68]

ASTER-4B 的模型输出（第15轮） [初始结果]

```
sum_y = sum(sols)
sum_y
```

执行结果：1/68 + 5*sqrt(185)/68

模型在此处从理论上找到了正确的和，但随后进入漫长的验证阶段，试图检查每个解的数值。

ASTER-4B 的模型输出（第 40 和 41 轮） [转折点：自我纠正]

```
# Manually checking the solution (2, 'A', -8/17)

n_val, seg, y_sym, x_sym = (2, 'A', -8/sp.Integer(17), 128/sp.Integer(17))
y_val=float(y_sym.evalf())
x_val=float(x_sym.evalf())
y_val, x_val
```

执行结果：(-0.470588..., 7.529411...)

分析： 现在进行数值_检查：$n = 2$，分段='A'，x 在 $[4(2) - 1,\ 4(2) + 1) = [7, 9)$。确实，$x = 7.529$ 在 $[7, 9)$ 之内。因此约束条件均已满足。但之前的数值检查将结果标记为 None，因为我们使用了 `floor(x/4)`... `floor(7.529/4)=1`。这导致结果被错误地判定为 $n = 1$。该解属于 $n = 2$。**结论：** 我之前的数值检查存在一个错误。全部 35 个解确实都有效。

5

ASTER-4B 在 AIME 2025 上的轨迹。这是 48轮交互的第三部分。

ASTER-4B 的模型输出（第 42 轮） [重新确认]

验证漏洞修复后，模型信任原先的 *35* 个解。

```
sum_y = sum([y_sym for *,*,y_sym,_ in solutions])
sum_y
```

执行结果：1/68 + 5*sqrt(185)/68

ASTER-4B 的模型输出（第 48 轮 - 最终）

坐标 y之和为 $\frac{1+5\sqrt{185}}{68}$。与形式 $\frac{a+b\sqrt{c}}{d}$相匹配：

- $a = 1$
- $b = 5$
- $c = 185$ (无平方因子， 5×37)
- $d = 68$

检查最大公约数：最大公约数（1、5、68） $= 1$。条件均已满足。计算
最终值： $a + b + c + d = 1 + 5 + 185 + 68 = 259$。最终答案是 259。

$\boxed{}$

6

Retool-4B 在 AIME 2025 上的轨迹。这是两轮交互的第一部分。

Retool-4B 的模型输出（第 1 轮） [初步分析]

（分别处理每条周期线段，然后求解由此得到的二次方程，并根据线段的 x 区间筛选解）.....因此，总体而言，对于 x *[0,34],*，我们有：*9* 个递增区间和 *9* 个递减区间，最后一个递减区间在 *34* 处截断。因此，我们需要为这 *9* 个递增区间和 *9* 个递减区间分别列出方程，并求解二次方程。不过，手动计算容易出错。让我们通过编程生成方程并求解，以找出每个区间内有效的解。

```
import math
# Define the intervals and solve quadratics for each n in increasing and decreasing
    cases

# Increasing intervals: defined by n from 0 to8
def solve_increasing():
    solutions = []
    for n in range(0, 9):
        a = 34
        b = -1
        c = -4 * n
        discriminant = b**2 - 4*a*c
        sqrt_disc = math.sqrt(discriminant)
        y1 = (-b + sqrt_disc) / (2*a)
        y2 = (-b - sqrt_disc) / (2*a)
        solutions.append((n, y1, y2))
    return solutions

# Decreasing intervals: defined by n from0 to8 (last cut off at x=34)
def solve_decreasing():
    solutions = []
    for n in range(0, 9):
        a = 34
        b = 1
        c = - (4 * n + 2)
    discriminant = b**2 - 4*a*c
    sqrt_disc = math.sqrt(discriminant)
    y1 = (-b + sqrt_disc) / (2*a)
    y2 = (-b - sqrt_disc) / (2*a)
    solutions.append((n, y1, y2))
    return solutions

# Calculate the valid y coordinates
valid_y = []

# Process increasing intervals
inc_solutions = solve_increasing()
for n, y1, y2 in inc_solutions:




    # Adjust intervals for n=0
    if n == 0:
        valid_x_interval = (0, 1)
    else:
        valid_x_interval = (interval_start, interval_end)
```

7

Retool-4B 在 AIME 2025 上的轨迹。这是两轮交互的第二部分。

Retool-4B 生成的模型输出（第 1 轮） [初始分析]

```
    # Check each solution
    for y in [y1, y2]:
        if -1 <= y <= 1:
            x = 34 * y**2
            if valid_x_interval[0] <= x < valid_x_interval[1]:
                valid_y.append(y)

# Process decreasing intervals
dec_solutions = solve_decreasing()
for n, y1, y2 in dec_solutions:
    x1 = 34 * y1**2
    x2 = 34 * y2**2
    interval_start = 4*n + 1
    interval_end = 4*n + 3

    # Adjust for n=8 (truncated at34)
    if n == 8:
        valid_x_interval = (33, 34)
    else:
        valid_x_interval = (interval_start, interval_end)

    for y in [y1, y2]:
        if -1 <= y <= 1:
            x = 34 * y**2
            if valid_x_interval[0] <= x < valid_x_interval[1]:
                valid_y.append(y)

# Sum all valid y coefficients
sum_y = sum(valid_y)

# Approximate the sum to verify
approx_sum = 4 * (sqrt(2) - 1)  # Approximation from original thinking

print("Valid y-coordinates:", valid_y)
print("Sum of y-coordinates:", sum_y)
print("Approximate sum:", approx_sum)
```

执行结果：未定义名称 'sqrt'

Retool-4B 生成的模型输出（第 2 轮 - 最终）

此错误表明，代码尝试使用 *sqrt*，却忘记导入它。让我们通过将所有手动调用的 *sqrt* 替换为 *math.sqrt(n)* 来修正这一问题。另外，让我们重写符号计算。等等，但是在上面的代码中，我们已经使用了 *math.sqrt(discriminant)*。因此，问题或许出在其他地方。等等，.....

因此，在 *valid_y* 列表中，我们得到了所有区间累积的 y 值。.....

但根据我们之前的代码输出，总和为 *1.1469*.....

答案是 24。