摘要

The GPT-Academic program cannot find abstract section in this paper.

Seed-Thinking-v1.5: Advancing Superb Reasoning Models with Reinforcement Learning

ByteDance Seed

Full author list in Contributions

Abstract

We introduce Seed-Thinking-v1.5, capable of reasoning through thinking before responding, resulting in improved performance on a wide range of benchmarks. Seed-Thinking-v1.5 achieves 86.7 on AIME 2024, 55.0 on Codeforces and 77.3 on GPQA, demonstrating excellent reasoning abilities in STEM and coding. Beyond reasoning tasks, the method demonstrates notable generalization across diverse domains. For instance, it surpasses DeepSeek R1 by 8% in win rate on non-reasoning tasks, indicating its broader applicability. Compared to other state-of-the-art reasoning models, Seed-Thinking-v1.5 is a Mixture-of-Experts (MoE) model with a relatively small size, featuring 20 B activated and 200 B total parameters. As part of our effort to assess generalized reasoning, we develop two internal benchmarks, BeyondAIME and Codeforces, both of which will be publicly released to support future research.

Date: April 10, 2025

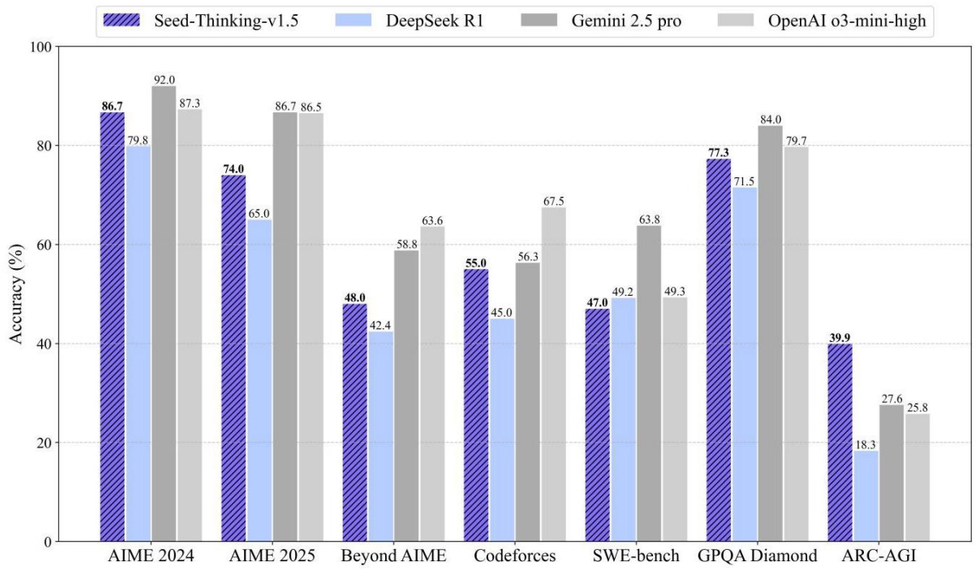


Figure 1 Benchmark performance on reasoning tasks

摘要

* 警告：该 PDF 由 GPT-Academic 开源项目调用大语言模型 +Latex 翻译插件一键生成，版权归原文作者所有。翻译内容可靠性无保障，请仔细鉴别并以原文为准。项目 Github 地址 https://github.com/binary-husky/gpt_academic/。当前大语言模型：qwen-plus，当前语言模型温度设定：0.2。为了防止大语言模型的意外谬误产生扩散影响，禁止移除或修改此警告。

GPT-Academic 程序无法在本文中找到摘要部分。

Seed-Thinking-v1.5: Advancing Superb Reasoning Models with Reinforcement Learning

字节跳动种子

贡献者名单中包含所有作者

Abstract

我们引入了 Seed-Thinking-v1.5，该模型能够在回应之前通过思考进行推理，从而在广泛的基准测试中表现出改进的性能。Seed-Thinking-v1.5 在 AIME 2024 上取得了 86.7 分，在 Codeforces 上取得了 55.0 分，在 GPQA 上取得了 77.3 分，展示了在 STEM 和编程方面的卓越推理能力。除了推理任务之外，该方法还在不同领域的多样化任务中表现出显著的泛化能力。例如，在非推理任务上，其胜率比 DeepSeek R1 高出 8%，表明其更广泛的应用潜力。与其他最先进的推理模型相比，Seed-Thinking-v1.5 是一个相对较小的专家混合（MoE）模型，具有 20 B 激活参数和 200 B 总参数。作为评估通用推理能力的一部分工作，我们开发了两个内部基准测试 BeyondAIME 和 Codeforces，并将它们公开发布以支持未来的相关研究。

日期：2025 年 4 月 10 日

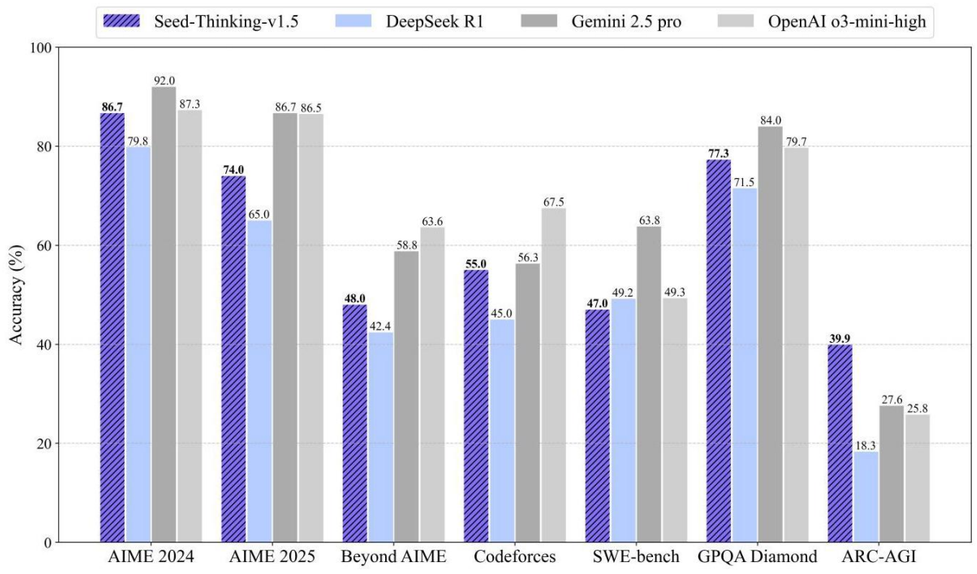


图 1 基准推理任务性能

1 Introduction

Driven by large-scale reinforcement learning on large language models, reasoning models have seen significant advancements. Notably, OpenAI's o1 series [1], DeepSeek's R1 [2], Google's Gemini 2.5 [3], and Anthropic's Claude 3.7 [4] have emerged as state-of-the-art models, each making substantial progress in logical reasoning, mathematical problem-solving, and code generation. These advancements underscore a shift toward more structured, efficient and scalable reasoning models, with ongoing research focusing on training efficiency, long chain-of-thought, and large-scale reinforcement learning.

In this work, we present a new reasoning model, called Seed-Thinking-v1.5. This model has achieved strong performance in both reasoning and non-reasoning tasks.

Mathematical Reasoning : For math competition, Seed-Thinking-v1.5 achieves 86.7 on AIME 2024, matching the performance of o3-mini-high and significantly outperforming o1 and DeepSeek R1, demonstrating competitive strength. Since AIME 2024 no longer provides sufficient discrimination, we construct a more challenging evaluation set named BeyondAIME. All problems in BeyondAIME are newly curated by human experts and designed to minimize the chance of being solved through memorization or guessing. While Seed-Thinking-v1.5 surpasses both o1 and R1, there remains a performance gap compared to o3 and Gemini pro 2.5. This also further demonstrates the discriminative power of the new evaluation set.

Competitive Programming : For the evaluation of competitive programming, we adopt Codeforces as our benchmark. Unlike some prior works that rely on Elo Scores, which contains estimation and are not directly comparable, we adopt a concrete evaluation protocol based on the most recent 12 Codeforces contests. Specifically, we report pass@1 and pass@8 metrics, where pass@k indicates whether the model solves the problem within k attempts, i.e., selecting the best result from k generated submissions. We choose to report pass@8 since it provides more stable results and aligns more closely with actual user submission patterns. Seed-Thinking-v1.5 outperforms DeepSeek R1 on both metrics, though a performance gap remains compared to o3. The evaluation set will be made publicly available in a future release.

Science : Seed-Thinking-v1.5 reached a score of 77.3 on GPQA, close to o3-level performance. Importantly, this gain is largely attributed to improved generalization from mathematical training, rather than an increase in domain-specific science data.

Non-reasoning Tasks : For non-reasoning tasks, Seed-Thinking-v1.5 is evaluated using a test set designed to replicate real-world user needs. Through human evaluations conducted against DeepSeek R1 across diverse scenarios, Seed-Thinking-v1.5 demonstrates significant advancements: it attains an 8.0% overall rise in users' positive feedback, thereby highlighting its augmented ability to manage intricate user scenarios.

There are three key points in the development of high-quality reasoning models: training data, RL algorithm, and RL infrastructure. We have devoted considerable effort to these three areas, and we will discuss them in detail.

Data For SFT training, unlike conventional post-training data, reasoning models rely on chain-of-thought data, which explicitly outlines the step-by-step reasoning process. Our preliminary experiments showed that too much non-CoT SFT data can significantly reduce the model's ability to explore. For RL training, we incorporate four categories of data: STEM problems, code-related tasks, logic reasoning and non-reasoning data like creative writing and dialogue. Among these, the logic reasoning data contributes to performance improvements on the ARC-AGI benchmark significantly. The math data exhibits strong generalization capabilities and can lead to broad performance improvements across tasks.

RL Algorithm RL training of reasoning models is highly unstable and often crashes, especially for models without SFT. Sometimes, the score difference between two runs can be as high as 10 points. The

1 Introduction

由大规模语言模型上的大规模强化学习驱动，推理模型取得了显著进展。特别是 OpenAI 的 o1 系列 [1]、DeepSeek 的 R1 [2]、Google 的 Gemini 2.5 [3] 和 Anthropic 的 Claude 3.7 [4] 已成为最先进的模型，每种模型在逻辑推理、数学问题解决和代码生成方面都取得了实质性进展。这些进步标志着向更加结构化、高效和可扩展的推理模型转变，当前的研究重点包括训练效率、长链推理和大规模强化学习。

在本文中，我们介绍了一种新的推理模型，称为 Seed-Thinking-v1.5。该模型在推理和非推理任务中均表现出色。

数学推理：在数学竞赛方面，Seed-Thinking-v1.5 在 AIME 2024 上取得了 86.7 分的成绩，与 o3-mini-high 的表现相当，并显著优于 o1 和 DeepSeek R1，展现出竞争力。由于 AIME 2024 不再提供足够的区分度，我们构建了一个更具挑战性的评估集，名为 BeyondAIME。BeyondAIME 中的所有问题均由人类专家全新策划，旨在尽量减少通过记忆或猜测解决问题的可能性。尽管 Seed-Thinking-v1.5 超越了 o1 和 R1，但与 o3 和 Gemini pro 2.5 相比仍存在性能差距。这也进一步证明了新评估集的区分能力。

竞技编程：对于竞技编程的评估，我们采用 Codeforces 作为基准。与一些依赖 Elo 分数（包含估计且不可直接比较）的先前工作不同，我们基于最近 12 场 Codeforces 比赛采用了具体的评估协议。具体来说，我们报告 pass@1 和 pass@8 指标，其中 pass@k 表示模型是否在 k 次尝试内解决问题，即从 k 次生成的提交中选择最佳结果。我们选择报告 pass@8，因为它提供了更稳定的结果，并更接近实际用户的提交模式。Seed-Thinking-v1.5 在这两项指标上均优于 DeepSeek R1，但与 o3 相比仍存在性能差距。评估集将在未来的发布中公开。

科学：Seed-Thinking-v1.5 在 GPQA 上得分为 77.3，接近 o3 水平的表现。重要的是，这一提升主要归功于数学训练带来的泛化能力增强，而非增加特定领域的科学数据。

非推理任务：对于非推理任务，Seed-Thinking-v1.5 使用一个旨在复制真实用户需求的测试集进行评估。通过对 DeepSeek R1 进行跨多种场景的人类评估，Seed-Thinking-v1.5 展现了显著的进步：它使用户的正面反馈总体提升了 8.0%，从而突显了其处理复杂用户场景的能力增强。

高质量推理模型的发展有三个关键点：训练数据、RL 算法和 RL 基础设施。我们在这些领域投入了大量努力，并将详细讨论它们。

数据对于 SFT 训练，与传统的后训练数据不同，推理模型依赖于链式思维数据，这种数据明确展示了逐步推理过程。我们的初步实验表明，过多的非 CoT SFT 数据会显著降低模型的探索能力。对于 RL 训练，我们纳入了四类数据：STEM 问题、与代码相关的任务、逻辑推理和非推理数据（如创意写作和对话）。其中，逻辑推理数据对 ARC-AGI 基准的性能提升贡献显著。数学数据表现出强大的泛化能力，可以带来广泛的任务性能提升。

RL 算法推理模型的 RL 训练高度不稳定，经常崩溃，尤其是对于没有经过 SFT 的模型。有时，两次运行之间的得分差异可能高达 10 分。RL 系统的稳定训练对推理模型的成功至关重要。为解决这些长期存在的问题，我们开创了 VAPO[5] 和 DAPO[6]——两种分别针对基于价值和无价值的 RL 范式的框架。VAPO 现已成为基于价值方法的最新技术（SOTA）解决方案，而 DAPO 则为无价值方法建立了新的 SOTA 结果。通过针对 RL 训练中的核心不稳定性问题，这两种方法提供了稳健和一致的训练轨迹，有效地实现了推理模型的可靠优化。RL 基础设施大型语言模型（LLM）为基础的强化学习系统的复杂性要求强大的基础设施，以确保可扩展性、可重复性和计算效率。为处理异构工作负载，我们采用了分离的流式 rollout 架构，该架构通过优先级样本池异步处理部分轨迹生成，相比同步框架实现了 3× 更快的迭代周期。该系统还支持混合精度训练并具备自动故障恢复功能，这对于在大规模 RL 运行期间保持稳定性至关重要。

stable training of RL systems is crucial for the success of reasoning models. To address these long-standing issues, we have pioneered VAPO[5] and DAPO[6]-two distinct frameworks tailored for value-based and value-free RL paradigms, respectively. VAPO now stands as the state-of-the-art (SOTA) solution in value-based methods, while DAPO establishes a new SOTA result for value-free approaches. By targeting the core instability issues in RL training, both methods deliver robust and consistent training trajectories, effectively enabling reliable optimization of reasoning models.

RL Infrastructure The complexity of Large Language Models (LLM) based reinforcement learning systems demands robust infrastructure to ensure scalability, reproducibility, and computational efficiency. To handle heterogeneous workloads, we decouple streaming rollout architecture that asynchronously processes partial trajectory generations through prioritized sample pools, achieving 3× faster iteration cycles than synchronous frameworks. The system also supports mixed-precision training with automatic fault recovery, critical for maintaining stability during large-scale RL runs.

2 Data

2.1 RL Training Data

Our RL training data consists of two main parts: verifiable problems with definitive answers and non-verifiable problems without definitive answers. The model's reasoning ability primarily comes from the first part and can be generalized to the second part.

2.1.1 Verifiable Problems

The Verifiable problems primarily comprise STEM questions paired with answers, coding problems equipped with unit tests, and logic reasonings that are amenable to automated verification.

STEM Data

Our dataset consists of several hundred thousand high-quality, competition-grade problems spanning mathematics, physics, and chemistry, with mathematics comprising the majority (over 80%). These problems are drawn from a mix of open-source datasets, public competitions (both domestic and international), and proprietary collections.

For data cleaning, we first eliminate questions with incomplete statements, inconsistent notation, or unclear requirements. For the remaining questions, we use our model (Doubao-Pro 1.5) to generate multiple responses. Problems for which the model achieved a woN score (worst of N) of 1 are deemed too simple and removed. Finally, some questions may have an inaccurate reference answer. We use SOTA reasoning models to generate multiple candidate responses for each question. If the model's answers were inconsistent with the reference answer, but the model's outputs showed high internal consistency, or involved only a very small number of reasoning tokens, we consider the reference answer to be incorrect. Human experts then conduct manual verification on these questions to ensure that the reference answers are correct. We also apply data augmentation to make the data more suitable for learning and evaluation. Specifically, we convert multiple-choice questions into fill-in-the-blank or short-answer formats to eliminate the possibility of guessing and to better assess reasoning ability. And we modify certain math problems to ensure that the answers are integers whenever possible.

2 Data

2.1 RL Training Data

我们的强化学习训练数据由两大部分组成：可验证的有确定答案的问题和不可验证的没有确定答案的问题。模型的推理能力主要来源于第一部分，并可以推广到第二部分。

2.1.1 Verifiable Problems

可验证问题主要包含配对有答案的 STEM 问题、带有单元测试的编码问题以及适合自动验证的逻辑推理。

STEM Data

我们的数据集包含数十万道高质量、竞赛级别的题目，涵盖数学、物理和化学领域，其中数学题目占大多数（超过 80%）。这些题目来源于开源数据集、公开竞赛（包括国内和国际）以及专有题库。

在数据清洗过程中，我们首先剔除陈述不完整、符号不一致或要求不清晰的题目。对于剩下的题目，我们使用我们的模型（Doubao-Pro 1.5）生成多个答案。对于模型达到 woN 分数（最差的 N 个分数）为 1 的题目，我们认为这些题目过于简单并将其移除。最后，某些题目可能具有不正确的参考答案。我们使用最先进的推理模型为每个问题生成多个候选答案。如果模型的答案与参考答案不一致，但模型的输出显示高度内部一致性，或者仅涉及少量推理标记，则我们认为参考答案是错误的。然后，由人类专家对这些问题进行人工验证，以确保参考答案正确。我们还应用数据增强技术，使数据更适合学习和评估。具体来说，我们将选择题转换为填空题或简答题形式，以消除猜测的可能性，并更好地评估推理能力。此外，我们修改了一些数学问题，尽可能确保答案为整数。

经过数据清洗和增强后，我们最终获得了一个包含 100k 道 STEM 问题的训练集。在训练过程中，我们使用基于模型的 Seed-Verifier 来评估答案的正确性，相关内容在 3.1 节中介绍。

Code Data

对于编码问题，我们优先选择高质量且具有挑战性的算法任务来源，主要来自备受尊敬的竞争性编程比赛。

我们对数据进行筛选，以确保每个问题都包含一个完整的规范：清晰的问题描述、一组单元测试和一个检查脚本。单元测试验证解决方案的功能正确性，而检查脚本则强制执行额外的约束条件，例如输出格式和边缘情况。我们还进行难度过滤，确保问题具备适当的复杂性和适用于现实世界的算法推理能力。

在评估方面，最准确的形式是将生成的代码提交到官方平台。然而，在强化学习过程中，实时提交并不可行。因此，我们开发了一个离线评估集，以便高效地进行本地验证。我们的观察表明，离线评估结果与官方评判结果之间存在很强的相关性。所有训练和评估问题都被整合到我们内部的代码沙箱环境中，从而可以直接执行和评估模型生成的代码。我们确保沙箱的稳定性和高吞吐量，以在 RL 训练过程中提供一致且准确的反馈。

Logical Puzzle Data

对于逻辑推理数据，我们收集了 22 个 commonly studied tasks，例如 24 点、迷宫、数独等。对于每个任务，我们构建了一个数据生成器和一个答案验证器。数据生成器可以自动产生大量的训练和评估数据。此外，对于许多任务，我们可以配置生成问题的难度。在训练过程中，我们会根据模型在某些任务上

After data cleaning and augmentation, we finally obtain a training set of 100k STEM problems. During training, we use model-based Seed-Verifier to evaluate response correctness, which is introduced in 3.1.

Code Data

For coding problems, we prioritize the source of high-quality and challenging algorithmic tasks, primarily drawn from esteemed competitive programming contests.

We filter data to ensure that each problem includes a comprehensive specification: a clear problem description, a set of unit tests, and a checker script. Unit tests validate the functional correctness of solutions, while the checker script enforces additional constraints such as output formatting and edge cases. We also perform difficulty filtering, ensuring that problems possess an appropriate level of complexity and applicability to real-world algorithmic reasoning.

For evaluation, the most accurate form is to submit the generated code to the official platforms. However, during reinforcement learning, real-time submission isn't feasible. Thus, we developed an offline evaluation set for efficient local validation. Our observations indicate a strong correlation between offline evaluation results and official verdicts. All training and evaluation problems are integrated into an in-house code sandbox environment, enabling direct execution and assessment of model-generated code. We ensure the sandbox's stability and high throughput to deliver consistent and accurate feedback during the RL training process.

Logical Puzzle Data

For the logic reasoning data, we gather 22 commonly studied tasks, such as 24-point, mazes, Sudoku, etc. For each task, we construct a data generator and an answer verifier. The data generator can automatically produce a large amount of training and evaluation data. Moreover, for many of the tasks, we can configure the difficulty of the generated problems. During the training process, we gradually adjust the difficulty of the training data based on the model's performance on certain tasks. The answer verifier rigorously evaluates the generation correctness and can be seamlessly integrated into RL pipelines as reward functions. We generate about 10 k puzzle problems for RL training.

2.1.2 Non-verifiable Problems

Non-verifiable problems mainly encompass non-reasoning tasks requiring quality assessment based on human preferences, involving tasks like creative writing, translation, knowledge QA, role-playing, and so on. The prompts are originated from RL training data for Doubao-1.5 Pro [7]. The dataset has sufficient coverage across diverse domains.

We discard data with low sample score variance and low difficulty. To be specific, we use the SFT model to generate multiple candidates for each prompt and then score them using a reward model. Prompts with low score variances are removed as they exhibit limited sampling diversity and minimal potential for improvement. Prompts are also removed where the reward score improvement surpasses a certain threshold during the Doubao 1.5 Pro RL training process [8]. This is because such data may be overly simplistic or already abundantly represented in the dataset. Offline experiments show that overoptimizing such samples leads to premature collapse of the model's exploration space and diminish the performance.

的表现逐步调整训练数据的难度。答案验证器严格评估生成结果的正确性，并且可以无缝集成到强化学习（RL）管道中作为奖励函数。我们生成了约 10 k 个谜题问题用于 RL 训练。

2.1.2 Non-verifiable Problems

不可验证的问题主要涵盖需要基于人类偏好进行质量评估的非推理任务，涉及创造性写作、翻译、知识问答、角色扮演等任务。提示词来源于 RL 训练数据集，用于通宝 1.5 Pro [7] 的训练。该数据集在不同领域中具有足够的覆盖率。

我们剔除了样本得分方差低且难度低的数据。具体来说，我们使用 SFT 模型为每个提示生成多个候选答案，并通过奖励模型对其进行评分。得分方差较低的提示被移除，因为它们表现出采样多样性有限且改进潜力较小的特点。此外，在通宝 1.5 Pro 的 RL 训练过程中 [8]，如果某个提示的奖励得分提升超过了某个阈值，也会被移除。这是因为这些数据可能过于简单或已经在数据集中过度表示。离线实验表明，对这些样本进行过度优化会导致模型探索空间的过早崩溃，并降低性能。

对于这些不可验证的数据，我们采用成对奖励方法进行评分和 RL 训练。通过比较两个样本之间的相对质量，这种方法有助于模型更好地理解用户偏好，从而提高生成结果的质量和多样性。奖励模型的详细信息将在 3.2 节中介绍。

2.2 Advanced Math Benchmark

当前的推理模型通常使用 AIME 作为评估数学推理能力的主要基准。然而，由于每年仅发布 30 道题目，其规模有限，可能导致评估结果的高方差，难以有效区分最先进的推理模型。为了更好地评估模型在数学推理方面的能力，我们构建了一个新的基准数据集：BeyondAIME。具体来说，我们与数学专家合作，根据已有的竞赛格式设计原创问题。我们通过结构修改和情景重构系统地改编现有的竞赛题目，确保没有直接复制的情况发生。此外，我们保证答案永远不会是简单的值——例如问题陈述中明确提到的数字——以减少模型在没有正确推理的情况下猜出正确答案的可能性。

通过这一严格的筛选和整理过程，我们最终编制了 100 道题目，每道题目的难度都等于或高于 AIME 中最难的问题。与 AIME 类似，所有答案都保证为整数（不局限于特定数值范围），这简化并稳定了评估过程。

3 Reward Modeling

作为强化学习（RL）中的一个关键组件，奖励建模定义了策略试图实现的目标或目的。因此，精心设计的奖励机制对于在训练阶段为模型响应提供准确且可靠的奖励信号至关重要。对于可验证问题和不可验证问题，我们采用不同的奖励建模方法。

Verifier-type	Training examples (approximate)	Human labeled testset
Seed-Verifier	$> 98\%$	82.7%
Seed-Thinking-Verifier	$> 99\%$	99.3%

表 1 两种验证器类型的准确性。具体来说，训练集上的准确性来源于训练统计结果。此外，我们手动标注了 456 个样本以形成测试集，这些样本特别选自 Seed-Verifier 无法稳定处理的情况。

3.1 Reward Modeling for Verifiable Problems

在正确的原则和思维轨迹下，我们利用大语言模型来判断各种场景下的大量可验证问题。这种方法产生了一个更通用的解决方案，超越了基于规则的奖励系统的限制。

For these non-verifiable data, we employ a pairwise rewarding method for scoring and RL training. By comparing the relative quality of two samples, this approach aids the model in better understanding user preferences, enhancing the quality and diversity of generated results. The detail of the reward model is introduced in 3.2.

2.2 Advanced Math Benchmark

The current reasoning models usually use AIME as the go-to benchmark to evaluate mathematical reasoning abilities. However, with only 30 problems released annually, its limited size can lead to high-variance evaluation results, making it challenging to effectively differentiate between state-of-the-art reasoning models. To better evaluate models' capabilities in mathematical reasoning, we construct a new benchmark dataset: BeyondAIME. Specifically, we collaborate with mathematics specialists to develop original problems informed by established competition formats. We systematically adapt existing competition questions through structural modifications and scenario reconfigurations, ensuring no direct duplication occurs. Furthermore, we ensure that the answers are never trivial values - such as numbers explicitly mentioned in the problem statement-to reduce the chance of models guessing the correct answer without proper reasoning.

Through this rigorous filtering and curation process, we compile a final set of 100 problems, each with a difficulty level equal to or greater than that of the hardest questions in AIME. Similar to AIME, all answers are guaranteed to be integers (without being restricted to a specific numerical range), which simplifies and stabilizes the evaluation process.

3 Reward Modeling

As a crucial component in RL, reward modeling defines the objective or goal that the policy is trying to achieve. Thus, a well-designed reward mechanism is essential to provide precise and reliable reward signals for model responses during the training stage. For verifiable and non-verifiable problems, we employ distinct reward modeling methodologies.

Verifier-type	Training examples (approximate)	Human labeled testset
Seed-Verifier	$> 98\%$	82.7%
Seed-Thinking-Verifier	$> 99\%$	99.3%

Table 1 Accuracy of two verifier-types. Specifically, the accuracy on the training set is derived from the training statistics. Additionally, we manually annotated 456 samples to form the test set, which are specifically selected from cases that the Seed-Verifier can not handle stably.

3.1 Reward Modeling for Verifiable Problems

With proper principles and thought trajectories, we utilize LLMs to judge a wide array of verifiable questions across diverse scenarios. This approach yields a more generalized solution that surpasses the limitations of rule-based reward systems.

We have designed two progressive reward modeling solutions, Seed-Verifier and Seed-Thinking-Verifier:

- Seed-Verifier is based on a set of meticulously crafted principles written by humans. It leverages the powerful foundational capabilities of LLMs to evaluate a triplet consisting of the question, reference

我们设计了两种渐进式的奖励建模解决方案：Seed-Verifier 和 Seed-Thinking-Verifier：

- Seed-Verifier 是基于一组由人类精心设计的原则构建的。它利用大语言模型的强大基础能力来评估一个问题、参考答案和模型生成答案所组成的三元组。如果参考答案和模型生成的答案在本质上是等价的，它将返回“YES”；否则返回“NO”。这里的等价性并非字面上的完全匹配，而是基于计算规则和数学原理的更深层次评估，以证明这两个答案具有相同的数学含义。这种方法确保了奖励信号能够准确反映模型的回答在本质上是否正确，即使表述方式有所不同。

- Seed-Thinking-Verifier 源于人类判断过程的启发，该过程通过细致的思考和深入的分析生成结论性的判断。为了实现这一点，我们训练了一个验证器，它为其评估提供了详细的推理路径。具体来说，我们将此视为一个可验证的任务，并与其他数学推理任务一起对其进行优化。该验证器可以剖析参考答案与模型生成答案之间的相似性和差异性，提供精确且细致的判断结果。

种子-思考-验证者显著缓解了与种子-验证者相关的三个主要问题：

- 奖励黑客行为:非思考型模型可能会利用漏洞,在没有真正理解问题的情况下获得奖励。Seed-Thinking-Verifier 中的详细推理过程使得这种黑客行为更加困难。

- 预测中的不确定性：在参考答案和模型生成的答案本质上等价的情况下，两者可能在格式上有所不同，例如 2^{19} 与 524288，Seed-Verifier 有时可能会返回“YES”，而有时返回“NO”。Seed-Thinking-Verifier 通过深入分析答案背后的推理过程，提供一致的结果。

- 边角案例的失败：存在一些 Seed-Verifier 难以有效处理的边缘案例。Seed-Thinking-Verifier 提供详细推理的能力使其能够更好地应对这些复杂场景。

表 1 展示了上述两种验证器的性能。更多关于案例研究的细节可以在附录 A 中找到。结果表明，Seed-Verifier 在有效处理某些特定情况方面存在困难，而 Seed-Thinking-Verifier 则表现出提供准确判断的卓越能力。尽管后者的思维过程确实消耗了大量 GPU 资源，我们认为它生成的精确且稳健的奖励结果对于赋予策略强大的推理能力至关重要。

3.2 Reward Modeling for Non-verifiable Problems

对于不可验证问题，我们训练了一个奖励模型用于强化学习（RL）训练。该奖励模型的训练数据与 Doubao 1.5 Pro [7] 中使用的人类偏好数据一致，主要涵盖创意写作和总结等类别。

为了提高奖励模型的效果，我们采用了 [9] 中提到的成对生成式奖励模型，该模型评估两个回答之间的优劣，并将“YES”或“NO”的概率作为最终的奖励分数。这种方法使得模型在评分时可以直接比较不同回答之间的差异，从而避免过度关注无关细节。实验结果表明，这种奖励建模方法提高了强化学习训练的稳定性，特别是在同时包含不可验证和可验证问题的混合训练场景中，通过最小化两种不同奖励建模范式之间的冲突来实现这一目标。这种改进可能归因于成对生成式奖励模型相较于传统奖励模型在缓解异常评分生成方面的固有优势，从而避免了与验证器之间评分分布的显著差异。

4 Approach

4.1 Supervised Fine-Tuning

我们的训练过程从监督微调（SFT）开始。SFT 阶段为后续的强化学习阶段奠定了坚实的基础。与从基础模型启动强化学习相比，SFT 模型生成的输出更易读，出现幻觉的情况更少，并且有害性也有所降低。我们整理了一个包含 400k 训练实例的 SFT 数据集，其中包括 300k 可验证问题和 100k 不可验证

answer, and model-generated answer. If the reference answer and model-generated answer are essentially equivalent, it returns "YES"; otherwise, it returns "NO". The equivalence here is not a literal exact match but rather a deeper assessment based on computational rules and mathematical principles that prove the two answers convey the same mathematical meaning. This approach ensures that the reward signal accurately reflects whether the model's response is correct in essence, even if the wording differs.

- Seed-Thinking-Verifier is inspired by the human judgment process, which generates conclusive judgments through meticulous thinking and in-depth analysis. To achieve this, we trained a verifier that provides a detailed reasoning path for its evaluations. Specifically, we treated this as a verifiable task and optimized it alongside other mathematical reasoning tasks. This verifier can dissect the similarities and differences between the reference and model-generated answers, offering precise and nuanced judgment results.

The Seed-Thinking-Verifier significantly alleviates three major issues associated with the Seed-Verifier:

- Reward Hacking: Non-thinking models may exploit loopholes to receive rewards without truly understanding the problem. The detailed reasoning process in Seed-Thinking-Verifier makes such hacking more difficult.

- Uncertainty in Predictions: In cases where the reference and model-generated answers are essentially equivalent, which may differ in format, e.g., 2^{19} vs 524288, the Seed-Verifier might sometimes return "YES" and other times "NO". The Seed-Thinking-Verifier provides consistent results by thoroughly analyzing the reasoning behind the answers.

- Failure on Corner Cases: There are certain edge cases that the Seed-Verifier struggles to handle effectively. The ability of Seed-Thinking-Verifier to provide detailed reasoning allows it to better address these complex scenarios.

Table 1 presents the performance of the above two verifiers. More details on case study can be found in Appendix A. The results indicate that the Seed-Verifier struggles to effectively handle some particular cases, whereas the Seed-Thinking-Verifier demonstrates a remarkable ability to provide accurate judgments. While the thinking process of the latter does consume a significant amount of GPU resources, we believe that the precise and robust reward results it generates are crucial for endowing the policy with strong reasoning capabilities.

3.2 Reward Modeling for Non-verifiable Problems

For non-verifiable problems, we train a reward model for RL training. The reward model training data is consistent with the human preference data utilized in Doubao 1.5 Pro [7], primarily encompassing categories such as creative writing and summarization.

To enhance the effectiveness of reward model, we adopt the pairwise generative reward model mentioned in [9]. which evaluates the superiority of two responses and use the probability of "YES" or "NO" as the final reward score. This approach enables the model to directly compare differences between responses during scoring, thereby avoiding excessive focus on irrelevant details. Experimental results demonstrate that this reward modeling method improves the stability of RL training, particularly in the

问题。可验证提示是从 RL 训练集中随机采样的。不可验证数据来源于用于 Doubao-Pro 1.5 [7] 的 SFT 数据，涵盖了创意写作、知识问答、安全性以及函数调用等领域。

为了生成高质量且带有长链推理（CoT）的响应，我们采用了一种集成模型合成、人工标注和拒绝采样的迭代工作流程。首先，人类专家通过提示工程技术或与内部模型进行交互对话，生成具有不同推理模式的响应。在积累数十个高质量的冷启动样本后，我们可以训练一个具备长 CoT 推理能力的模型，作为更强大的助手。然后，我们使用 Seed-Verifier 对该推理模型进行拒绝采样。尽管此工作流程主要应用于数学数据，但我们观察到它也可以很好地推广到其他领域，例如编程、逻辑谜题，甚至创意写作。因此，在其他领域中，我们也进行了冷启动过程并结合拒绝采样来生成详细的推理轨迹。

在训练过程中，每个实例被截断为 32,000 个标记。我们使用上述数据对基础模型进行两次 epoch 的微调。我们采用余弦退火学习率调度策略，其中峰值学习率（lr）为 2×10^{-5}，并逐渐衰减至 2×10^{-6}。

4.2 Reinforcement Learning

我们已经开发了一个统一的强化学习框架，该框架可以无缝融合来自广泛领域的需求数据。这一整合包含了三个数据类别：

- 可验证数据，从验证者那里获得反馈。这种类型的数据允许根据已知标准直接验证模型的输出。

- 通用数据，由奖励模型评分。奖励模型根据模型的响应与人类偏好的一致性程度分配分数。

- 一类特定的数据集，结合了验证者和奖励模型的分数。这种混合数据类型利用了验证和基于奖励的评估的优势。

在长链推理强化学习人类反馈（long-CoT RLHF）的背景下，我们面临几个挑战，例如价值模型偏差和奖励信号的稀疏性。为了解决这些问题，我们借鉴了之前工作中的关键技术 [5, 6, 10] ：

- 值预训练：我们从一个固定的策略（例如 π_{sft} ）中采样响应，并使用蒙特卡洛回报更新值模型。这一过程确保了初始化的值模型完全与我们的策略 π_{sft} 对齐。保持这种对齐已被证明对于保留模型的链式思维（CoT）模式至关重要，从而使模型能够生成连贯且逻辑清晰的链式思维。

- 解耦-GAE：通过使用不同的广义优势估计（GAE）参数，例如 $\lambda_{\text{value}} = 1.0$ 和 $\lambda_{\text{policy}} = 0.95$，我们允许价值模型以无偏的方式进行更新。同时，策略可以独立地平衡自身的偏差和方差。这种解耦合使得模型的训练更加高效和稳定。

- 长度自适应 GAE：我们设定 $\lambda_{\text{policy}} = 1 - \frac{1}{\alpha l}$ ，其中 α 是一个超参数，l 是响应长度。这种方法确保了在短序列和长序列之间时间差分（TD）误差的分布更加均匀。因此，模型在训练过程中能够更有效地处理不同长度的序列。

- Clip-Higher：在近端策略优化（PPO）算法中，我们如下所述地将上下剪切界限解耦：

$$\mathcal{L}^{CLIP}(\theta) = \widehat{\mathbb{E}}_t \left[\min \left(r_t(\theta) \widehat{A}_t, \text{clip}\left(r_t(\theta), 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}}\right) \widehat{A}_t \right) \right] \tag{1}$$

通过增加 ϵ_{high} 的值，我们为低概率词元的增加创造了更多空间。这鼓励模型探索更广泛的可能响应范围，增强其发现新颖且有效解决方案的能力。

- Token-level Loss: 我们不是在整个响应上定义策略损失，而是在所有标记上定义它。这种方法解决了标记对最终损失的贡献不平衡的问题，确保每个标记对训练过程的影响得到适当的考虑。

- 正例语言模型损失（Positive Example LM Loss）：该损失函数旨在提升强化学习（RL）训练过程中正例样本的利用效率。我们为正例添加了一个带系数 μ 的语言模型损失：

mixed training scenarios involving both non-verifiable and verifiable problems, by minimizing conflicts between the two different types of reward modeling paradigms. This improvement may be attributed to the pairwise generative reward model's inherent advantage in mitigating outlier score generation compared to conventional reward models, therefore avoiding significant discrepancies in score distributions with the verifier.

4 Approach

4.1 Supervised Fine-Tuning

Our training process starts with supervised fine-tuning (SFT). The SFT phase sets a solid foundation for the subsequent reinforcement learning stage. Compared to initiating RL from a base model, the SFT model produces more readable outputs, exhibits fewer instances of hallucination, and demonstrates reduced harmfulness. We curate an SFT data comprising 400k training instance, including 300k verifiable problems and 100k non-verifiable problems. Verifiable prompts are randomly sampled from RL training set. Non-verifiable data are sourced from the SFT data used for Doubao-Pro 1.5 [7], covering areas such as creative writing, knowledge-based QA, safety, and function calling.

To generate high-quality responses with long CoT, we employ an iterative workflow that integrates model synthesis, human annotation, and rejection sampling. Initially, human experts apply prompt engineering techniques or engage in interactive dialogues with an internal model to produce responses with various reasoning patterns. After accumulating tens of high-quality cold-start samples, we can train a reasoning model with long CoT as a more capable assistant. Then we perform rejection sampling on this reasoning model using Seed-Verifier. While this workflow is primarily applied to mathematical data, we observe it can generalize well to other domains, such as coding, logic puzzle and even creative writing. Thus, for other domains, we also conduct a cold start process followed by rejection sampling to produce detailed reasoning trajectories.

During training, each instance is truncated to 32,000 tokens. We fine-tune the base model for two epochs using the above data. We use a cosine decay learning rate scheduling that the peak lr is 2×10^{-5} and decays to 2×10^{-6} gradually.

4.2 Reinforcement Learning

We have developed a unified reinforcement learning framework that seamlessly fuses data from a broad range of domains. This integration incorporates three data categories:

- Verifiable data, which obtains feedback from a verifier. This type of data allows for direct validation of the model's outputs against known criteria.
- General data, scored by a reward model. The reward model assigns scores based on how well the model's responses align with human preferences.
- A specific class of data that combines scores from both the verifier and the reward model. This hybrid data type leverages the strengths of both verification and reward-based evaluation.

In the context of long-CoT RLHF, we encounter several challenges such as value model bias and the sparsity of reward signals. To address these issues, we draw on key techniques from our prior work [5, 6, 10] :

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{PPO}}(\theta) + \mu * \mathcal{L}_{\text{NLL}}(\theta) \tag{2}$$

这一附加的损失项帮助模型更好地从正例中学习，从而提高其整体性能。

在合并来自不同领域的数据并纳入多样化的评分机制时，我们面临着不同数据领域之间干扰的挑战。这种干扰可能源于难度水平的差异、奖励劫持的风险以及其他潜在因素。这些问题使得在整个模型的所有能力上实现均匀和同时的改进变得极为困难。为了解决这一问题，我们引入了在线数据分布适应（Online Data Distribution Adaptation）。该方法将强化学习过程中静态的提示分布转换为更适应模型训练需求的动态分布。通过这种方式，我们减少了数据干扰带来的负面影响，并确保在不同能力上的更均衡改进。因此，模型能够在广泛的任务中更一致地提升其性能。

5 Infrastructures

5.1 Framework

训练框架是使用 HybridFlow [11] 编程抽象构建的。整个训练工作负载运行在一个 Ray [12] 集群之上。数据加载器和强化学习（RL）算法在一个单进程 Ray Actor（单控制器）中实现。模型训练和响应生成（rollout）则在 Ray Worker Group 中实现。Ray Worker Group 暴露了一组 API（例如，generate_response/train_batch 等），这些 API 通过 SPMD（单程序多数据）在 Worker Group 内部执行繁重的训练/生成任务。单控制器调用 Ray Worker Group 暴露的各种 API 来构建训练流程。HybridFlow 编程抽象使得快速原型化强化学习算法思想成为可能，而无需担心复杂的分布式系统。

Seed-Thinking-v1.5 是通过混合引擎架构 [13] 训练的，其中所有模型都共存于同一位置。这可以避免在训练和生成之间切换时 GPU 的空闲时间。在 Long-CoT 生成过程中，我们观察到由于不同提示之间的响应长度差异较大而导致的严重拖尾现象。这在生成过程中造成了大量的 GPU 空闲时间。为缓解长尾响应生成中的拖尾问题，我们提出了 SRS（Streaming Rollout System）——一种资源感知调度框架，该框架战略性地部署独立的流式计算单元，将系统约束从内存限制转变为计算限制。

5.2 Streaming Rollout System

SRS 架构引入了流式展开，以解耦模型演化与运行时执行，通过参数化 α 实现开环/闭环样本比例的动态调整：

- 定义完成率 ($\alpha \in [0, 1]$) 为使用最新模型版本通过策略生成的样本比例。
- 将剩余的非完整段（1- α ）分配给来自版本化模型快照的离策略回滚，通过在独立资源上异步继续部分生成来无缝集成。

此外，我们在环境交互阶段还实现了动态精度调度，通过带有误差补偿范围缩放的训练后量化部署 FP8 策略网络。为了解决 MoE 系统中的 token 不平衡问题，我们实现了一个三层并行架构，结合了层计算的 TP（张量并行）、具有动态专家分配的 EP（专家并行）以及用于上下文分块的 SP（序列并行）。我们的内核自动调优器根据实时负载监控动态选择最优的 CUDA 内核配置。

5.3 Training System

为了高效地大规模训练 Seed-Thinking-v1.5 模型，我们设计了一个混合分布式训练框架，该框架集成了先进的并行策略、动态工作负载平衡和内存优化。以下我们将详细介绍推动系统效率和可扩展性的核心技术创新。

- Value-Pretraining: We sample responses from a fixed policy, such as π_{sft} , and update the value model using the Monte-Carlo return. This process ensures that the initialized value model is fully aligned with our policy π_{sft} . Maintaining this alignment has been proven to be crucial for preserving the model' s CoT pattern, enabling the model to generate coherent and logical CoT.

- Decoupled-GAE: By employing different Generalized Advantage Estimation (GAE) parameters, such as $\lambda_{\text{value}} = 1.0$ and $\lambda_{\text{policy}} = 0.95$, we allow the value model to update in an unbiased manner. Meanwhile, the policy can independently balance its own bias and variance. This decoupling enables more efficient and stable training of the model.

- Length-adaptive GAE: We set $\lambda_{\text{policy}} = 1 - \frac{1}{\alpha l}$, where α is a hyper-parameter and l is the response length. This approach ensures a more uniform distribution of Temporal Difference (TD) errors across both short and long sequences. As a result, the model can handle sequences of varying lengths more effectively during training.

- Clip-Higher: In the Proximal Policy Optimization (PPO) algorithm, we decouple the upper and lower clip bounds as follows:

$$\mathcal{L}^{CLIP}(\theta) = \widehat{\mathbb{E}}_t \left[\min \left(r_t(\theta) \widehat{A}_t, \text{clip}\left(r_t(\theta), 1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}}\right) \widehat{A}_t \right) \right] \tag{1}$$

By increasing the value of ϵ_{high} , we create more room for the increase of low-probability tokens. This encourages the model to explore a wider range of possible responses, enhancing its ability to discover novel and effective solutions.

- Token-level Loss: Instead of defining the policy loss over entire responses, we define it over all tokens. This approach addresses the imbalance in the token-level contribution to the final loss, ensuring that each token's impact on the training process is appropriately accounted for.

- Positive Example LM Loss: This loss function is designed to boost the utilization efficiency of positive samples during the RL training process. We add a language model loss with a coefficient μ for positive examples:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{PPO}}(\theta) + \mu * \mathcal{L}_{\text{NLL}}(\theta) \tag{2}$$

This additional loss term helps the model to better learn from positive examples, improving its overall performance.

When merging data from different domains and incorporating diverse scoring mechanisms, we face the challenge of interference between different data domains. This interference can arise from disparities in difficulty levels, the risk of reward-hacking, and other underlying factors. These issues make it extremely difficult to achieve uniform and simultaneous improvements across all capabilities of the model. To counteract this, we introduce Online Data Distribution Adaptation. This method transforms the stationary prompt distribution during reinforcement learning into an adaptive distribution that better caters to the model's requirements during training. By doing so, we minimize the negative impact of data interference and ensure a more balanced improvement across different abilities. As a result, the model can enhance its performance more consistently across a wide array of tasks.

- 并行机制。我们结合张量并行（TP）/ 专家并行（EP）/ 上下文并行（CP）与全分片数据并行（FSDP）来训练 Seed-Thinking-v1.5。具体来说，我们在注意力层应用了 TP/CP，并在 MoE 层应用了 EP。
- 序列长度平衡。有效的序列长度在 DP 等级之间可能不平衡，导致计算工作量不平衡，训练效率低下。为了解决这一挑战，我们采用了 KARP [14] 算法，该算法重新排列一个小批量（mini-batch）中的输入序列，以使它们在微批次（micro-batches）之间保持平衡。
- 内存优化。我们采用逐层重新计算 [15]、激活卸载和优化器卸载来支持更大的微批次训练，以重叠由 FSDP 引起的通信开销。
- 自动并行性。为了实现最佳系统性能，我们开发了一个自动调优系统，称为 AutoTuner。具体来说，AutoTuner 基于配置文件的解决方案 [16] 对内存使用情况进行建模。然后，它估计不同配置的性能和内存使用情况，以获得最佳配置。
- 检查点。我们采用 ByteCheckpoint [17] 来支持从不同的分布式配置中以最小的开销恢复检查点。这使得用户能够弹性地训练任务，从而提高集群效率。

基准测试	Seed-Thinking-v1.5	DeepSeek R1	OpenAI o3-mini	Grok 3 Beta	Gemini 2.5 pro
数学					
AIME 2025	74.0%	65.0%	86.5%	77.3%	86.7%
AIME 2024	86.7%	79.8%	87.3 %	83.9%	92.0%
超越 AIME	48.0%	42.4%	63.6 %	-	58.8%
科学					
GPQA 钻石	77.3%	71.5%	79.7%	80.2%	84.0%
SuperGPQA	62.1%	60.5%	52.2%	62.8%	65.3%
MMLU-PRO	87.0%	85.6%	82.4%	84.6%	86.3%
代码					
Codeforces avg@8	36.3%	32.0%	50.9%	-	40.3%
Codeforces pass@8	55.0%	45.0%	67.5%	-	56.3%
LiveCodeBench v5	64.9%	64.3%	74.1%	70.6%	70.4%
Aider Polyglot	54.2%	56.9%	68.6%	-	74.0%
代理编码					
SWE-bench 已验证	47.0%	49.2%	49.3%	-	63.8%
SWE-bench 已验证 *	47.0%	46.2%	44.5%	-	63.8%
逻辑推理					
ARC-AGI	39.9%	18.3%	25.8%	31.9%	27.6%
事实性					
SimpleQA	12.9%	30.1%	13.8%	43.6%	52.9%
指令					
Collie	73.1%	34.2%	87.6%	33.6%	62.5%
IFEval	87.4%	86.1%	93.7%	83.4%	91.5%

表 2 最先进的推理模型的结果

* 结果来自我们的内部沙盒，由于测试环境的不一致，可能与报告的结果有所不同。

6 Experiment Results

6.1 Auto Evaluation Results

表 2 展示了跨数学、编程、科学和常识领域多样化任务的评估结果。对于数学基准任务，结果是根据 32 个模型响应的平均值计算得出的，而 GPQA 任务的结果则是基于 8 个响应的平均值。对于 Codeforces，

5 Infrastructures

5.1 Framework

The training framework is built using HybridFlow [11] programming abstraction. The whole training workload runs on top of a Ray [12] cluster. The dataloader and RL algorithm is implemented in a single process Ray Actor (single controller). The model training and response generation (rollout) is implemented in a Ray Worker Group. The Ray Worker Group exposes a set of APIs (e.g., generate_response/train_batch, etc,.), which runs heavy training/generation workload via SPMD (single program, multiple data) inside the Worker Group. The single controller invokes various APIs exposed by the Ray Worker Group to construct the training flow. HybridFlow programming abstraction enables fast prototyping of RL algorithm ideas without bothering with complex distributed systems.

Seed-Thinking-v1.5 is trained through hybrid engine architecture [13], where all the models are co-located. This prevents the idle time of the GPUs when switching between training and generation. During Long-CoT generation, we observe severe straggler phenomenon caused by the large difference of the response length between various prompts. This causes massive GPU idle time during generation. To mitigate the straggler of long-tail response generation, we propose SRS (Streaming Rollout System) - a resource-aware scheduling framework that strategically deploys standalone streaming-compute units to transform system constraints from memory-bound to compute-bound.

5.2 Streaming Rollout System

The SRS architecture introduces streaming rollout to decouple model evolution from runtime execution, enabling dynamic adjustment of on/off-policy sample ratios through parametric α :

- Define the completion ratio ($\alpha \in [0, 1]$) as the proportion of samples generated on-policy using the latest model version
- Allocate the remaining non-complete segment (1- α) to off-policy rollouts from versioned model snapshots, seamlessly integrated through asynchronous continuation of partial generations on the standalone resources.

In addition, we also implement dynamic precision scheduling during environment interaction phases, which deploys FP8 policy networks via post-training quantization with error-compensated range scaling. To address token imbalance in MoE systems, we implement a three-tiered parallel architecture combining TP (tensor parallelism) for layer-wise computation, EP (expert parallelism) with dynamic expert assignment, and SP (sequence parallelism) for context chunking. Our kernel auto-tuner dynamically selects optimal CUDA kernel configurations based on real-time load monitoring.

5.3 Training System

To efficiently train the Seed-Thinking-v1.5 model at scale, we design a hybrid distributed training framework that integrates advanced parallelism strategies, dynamic workload balancing, and memory optimizations. Below we detail the core technical innovations driving the system's efficiency and scalability.

- Parallelism mechanisms. We compose TP (tensor parallelism)/EP (expert parallelism)/CP (context parallelism) with Fully Sharded Data Parallelism (FSDP) to train Seed-Thinking-v1.5. Specifically, we applied TP/CP for attention layers, and EP for MoE layers.

我们同时报告了 avg@8 和 pass@8，因为 pass@8 更符合人类提交的习惯。所有其他任务的结果均基于 1 个响应的平均值。

在数学推理方面，Seed-Thinking-v1.5 在 AIME 2024 基准上表现出顶级性能，得分为 86.7，与 OpenAI 的 o3-mini-high 模型性能相当。然而，在更近期的 AIME 2025 以及高级的 BeyondAIME 挑战中，Seed-Thinking-v1.5 仍然落后于 o3 级别的表现。对于 GPQA 任务，Seed-Thinking-v1.5 达到了 77.3% 的准确率，接近 o3-mini-high 的性能。在如 Codeforces 之类的代码生成场景中，Seed-Thinking-v1.5 几乎与 Gemini 2.5 Pro 的表现持平，但仍然逊色于 o3-mini-high。值得注意的是，Seed-Thinking-v1.5 在 SimpleQA 上的表现不够出色。需要强调的是，这一基准主要作为一个记忆导向的指标，其性能与预训练模型的规模相关性更强，而非真实的推理能力。

6.2 Human Evaluation Results

为了评估模型在主观任务上的表现，其中自动化指标不足以捕捉微妙的人类偏好，我们在一系列多样的非推理场景中进行人工评估。我们的评估旨在衡量质量的关键维度，例如连贯性、相关性、创造力以及对以人类为中心的偏好的遵循程度。由领域专家组成的评审小组根据预定义的标准，对模型输出与 Deepseek R1 进行评分。我们使用一个 5 分的序数尺度，从 0（非常差）到 4（优秀），并在包含多轮对话的会话提示下评估两个模型。每个完整的会话都会用二元胜负结果标注，以捕捉整体用户体验，并且每一轮都会分配一个 0-4 的单一分数。

Seed-Thinking-v1.5 在被评估的会话中实现了 8.0% 的整体胜率，表明其在符合以人类为中心的偏好方面具有优越性。此外，这一胜率在各种场景中保持一致，从创意写作到人文知识的阐述均如此。图 2 显示了每轮得分的分布情况。

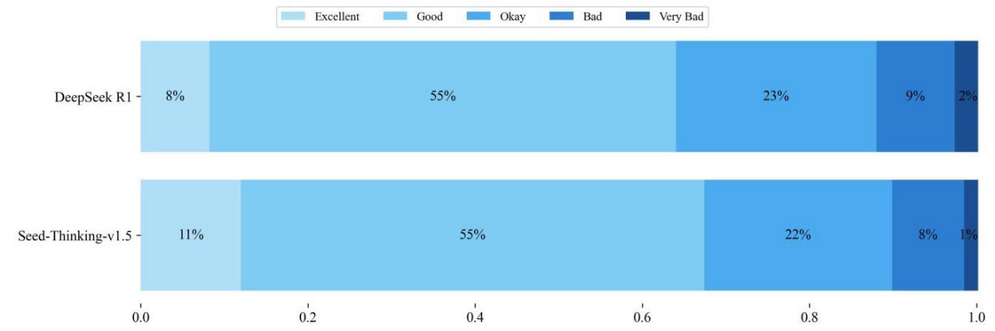


图 2 评分分布

6.3 Effects of pre-train models

拒绝采样。拒绝采样已被确定为一种提高模型性能的有价值技术 [2]。我们进行了一项消融研究，以检查使用拒绝微调（RFT）模型初始化强化学习是否会影响结果。我们的结果显示，使用 RFT 初始化的预训练模型在训练过程中饱和得更快，但最终性能低于未使用 RFT 训练的模型，如表 3 所示。

跨模型规模的一致算法排名。我们观察到强化学习算法在不同规模和架构的模型中表现出一致的排名行为。如表 4 所示，Seed-150B-MoE 是一个与 Qwen-32B 在架构（MoE 对比密集）和规模上都不同的模型，却表现出一致的排名。值得注意的是，这种一致性表明 Qwen-32B 可以有效地作为研究强化学习算法的代理模型。

- Sequence length balancing. The effective sequence length can be imbalanced across DP ranks, leading to imbalanced computation workload and low training efficiency. To address this challenge, we leverage KARP [14] algorithm that rearranges the input sequences within one mini-batch to make them balance among micro-batches.

- Memory optimization. We adopt layer-wise recomputation [15], activation offload and optimizer offload to support training of larger micro-batches to overlap the communication overhead caused by FSDP.

- Auto parallelism. To enable optimal system performance, we develop an automatic tuning system, referred to as AutoTuner. Specifically, AutoTuner models the memory usage following a profile-based solution [16]. Then, it estimates the performance and memory usage of various configurations to obtain the optimal configuration.

- Checkpoint. We employ ByteCheckpoint [17] to support checkpoint resume from different distributed configurations with minimal overhead. This enables users to elastically train the tasks to improve cluster efficiency.

Benchmark	Seed-Thinking-v1.5	DeepSeek R1	OpenAI o3-mini	Grok 3 Beta	Gemini 2.5 pro
Mathematics					
AIME 2025	74.0%	65.0%	86.5%	77.3%	86.7%
AIME 2024	86.7%	79.8%	87.3 %	83.9%	92.0%
Beyond AIME	48.0%	42.4%	63.6 %	-	58.8%
Science					
GPQA diamond	77.3%	71.5%	79.7%	80.2%	84.0%
SuperGPQA	62.1%	60.5%	52.2%	62.8%	65.3%
MMLU-PRO	87.0%	85.6%	82.4%	84.6%	86.3%
Code					
Codeforces avg@8	36.3%	32.0%	50.9%	-	40.3%
Codeforces pass@8	55.0%	45.0%	67.5%	-	56.3%
LiveCodeBench v5	64.9%	64.3%	74.1%	70.6%	70.4%
Aider Polyglot	54.2%	56.9%	68.6%	-	74.0%
Agentic Coding					
SWE-bench verified	47.0%	49.2%	49.3%	-	63.8%
SWE-bench verified *	47.0%	46.2%	44.5%	-	63.8%
Logic reasoning					
ARC-AGI	39.9%	18.3%	25.8%	31.9%	27.6%
Factuality					
SimpleQA	12.9%	30.1%	13.8%	43.6%	52.9%
Instruction					
Collie	73.1%	34.2%	87.6%	33.6%	62.5%
IFEval	87.4%	86.1%	93.7%	83.4%	91.5%

Table 2 Results of State-of-the-Art Reasoning Models

*Results from our internal sandbox, which may differ from the reported results due to inconsistencies in the testing environment.

Models	AIME avg@32
Baseline	58%
w/ RFT	54%

Table 3 Ablations on Pretrained Models

AIME	DAPO	VAPO
Qwen-32B-Dense	50%	60%
Seed-150B-MoE	73%	79%

表 4 一致的算法排名。Seed-150B- MoE 的结果仅限步数的消融研究。

7 Related Work

测试时扩展 [4, 18-20]，例如 OpenAI 的 o1 [1] 和 DeepSeek 的 R1 [2]，推动了大语言模型 (LLMs) 的深刻范式转变 [21, 22]。通过实现扩展的链式思维 (CoT) 推理 [23] 并激发复杂的推理能力，这些方法使 LLMs 在复杂的数学和编程任务中表现出色，包括来自 AIME 和 Codeforces 等竞赛的任务。这一转型的核心是大规模强化学习，它促进了复杂推理行为的出现——例如自我验证和迭代改进。然而，支持可扩展强化学习训练的关键方法和算法在很大程度上仍然模糊不清，通常被排除在现有推理模型的技术文档之外 [1, 2, 21-23]。在本文中，我们介绍了一个达到 SOTA 水平的模型 Seed-Thinking-v1.5，并从三个方面详细介绍其实现性能的过程：数据、强化学习算法和强化学习基础设施。

8 Conclusion

我们引入了一种卓越的推理模型，名为 Seed-Thinking-v1.5，该模型在推理任务和非推理任务中均表现出色。它通过使用先进的强化学习（RL）技术，在 AIME24 上达到 86.7%，AIME25 上达到 74.0%，Codeforces 上达到 55.0%，从而稳定且可靠地提升其思维能力。未来，我们计划研究更高效的强化学习方法，并通过思维模式探索更具挑战性的任务，以拓展模型智能的边界。此外，具有与验证器相当精度的通用奖励建模也将是一个极具吸引力的研究方向。

Contributions and Acknowledgments

Core Contributors

（作者姓名随机打乱）

袁玉峰
岳禹
王明轩
左晓辰
陈佳泽
颜琳
徐文媛
张驰
刘欣
王成一
范天天

6 Experiment Results

6.1 Auto Evaluation Results

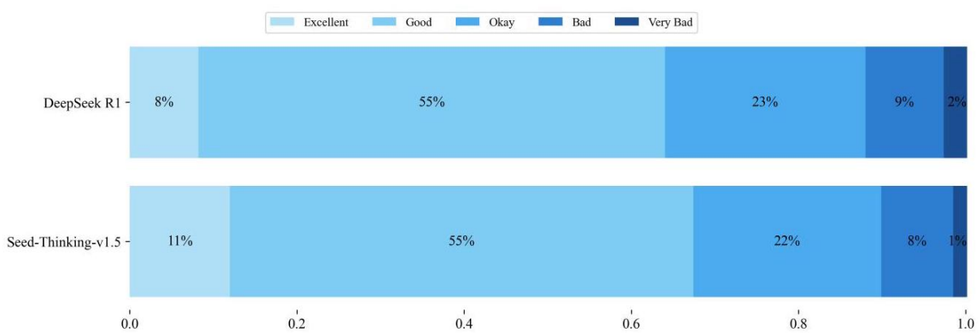
Table 2 presents the evaluation results across diverse tasks spanning mathematics, coding, science, and general knowledge domains. For mathematical benchmark tasks, results are calculated as the average across 32 model responses, while GPQA task results are averaged over 8 responses. For Codeforces, we report both avg@8 and pass@8, because pass@8 aligns better with human submission habits. Results for all other tasks are averaged over 1 response.

In mathematical reasoning, Seed-Thinking-v1.5 achieves top-tier performance on the AIME 2024 benchmark, scoring 86.7, matching the performance of OpenAI's o3-mini-high model. However, on the more recent AIME 2025 and the advanced BeyondAIME challenges, Seed-Thinking-v1.5 still lags behind o3-level performance. For the GPQA task, Seed-Thinking-v1.5 achieves an 77.3% accuracy rate, close to the performance of o3-mini-high. In code generation scenarios such as Codeforces, Seed-Thinking-v1.5 nearly matches the performance of Gemini 2.5 Pro but still trails behind o3-mini-high. Notably, Seed-Thinking-v1.5 demonstrates less impressive results on SimpleQA. It is worth emphasizing that this benchmark primarily functions as a memory-oriented metric, where performance is more strongly correlated with pre-trained model scale rather than genuine reasoning capabilities.

6.2 Human Evaluation Results

To evaluate model performance on subjective tasks, where automated metrics are insufficient to capture nuanced human preferences, we conduct human evaluations across a diverse suite of non-reasoning scenarios. Our assessments are designed to measure key dimensions of quality, such as coherence, relevance, creativity, and adherence to human-centric preferences, with a panel of domain-expert evaluators rating model outputs against Deepseek R1 under predefined rubrics. We use a 5-point ordinal scale, ranging from 0 (very poor) to 4(excellent), and evaluate both models on session prompts with multiple rounds. Each full session is annotated with a binary win/loss outcome to capture the overall user experience and a single 0-4 score is assigned per-round.

Seed-Thinking-v1.5 achieves an overall win ratio of 8.0% on the evaluated sessions, indicating superiority in aligning with human-centric preferences. Further more, this win rate is consistent across diverse scenarios, from creative writing to humanities knowledge elaboration. Figure 2 shows the per-round level score distribution.



刘灵军
余启颖
魏翔鹏
林智琦

Contributors

朱若飞
杨清平
魏成志
何杰
刘冠林
吴征 *
余翔宇
刘志成
徐晶晶
陈江杰
潘浩杰
胡圣丁 *
杜正银
王文琦
孙泽伟
楼晨炜
马博乐
王梓涵
张莫凡
张旺
刘高鸿
蒋凯华
林海斌
张茹
刘军才
韩莉
迟金鑫

Pretraining Efforts

张文强徐佳怡袁军肖振昊鲜宇桥吴景桥汪凯华娜周建辉段何阳鲁昌宝王金祥欧世航王晓然金学松姚成银徐文昌马哲诚安任明庞小肖靖苏瑜瑜张涛孙凯波刘义凡孙一凡沈思君张义元马兴妍卓李姚罗德毅刘世一湛云水李元杨德发朱科沈成刚李训周亮向理

Team Lead

吴永辉

Figure 2 Rating Distribution

6.3 Effects of pre-train models

Rejection Sampling. Rejection sampling has been identified as a valuable technique for improving model performance [2]. We perform an ablation to examine whether initializing RL with a rejection fine-tuning (RFT) model impacts outcomes. Our results show that the pretrained model initialized with RFT saturates more quickly during training but ultimately achieves lower performance than the model trained without RFT, as shown in Table 3.

Consistent algorithm rankings across model size. We observe that RL algorithms demonstrate consistent ranking behaviors across different models of varying sizes and architectures. As illustrated in Table 4, Seed-150B-MoE, a model that differs from Qwen-32B in both architecture (MoE vs. dense) and size, exhibits a consistent ranking. Notably, this consistency suggests that Qwen-32B can effectively serve as a proxy model for investigating RL algorithms.

Models	AIME avg@32
Baseline	58%
w/ RFT	54%

Table 3 Ablations on Pretrained Models

AIME	DAPO	VAPO
Qwen-32B-Dense	50%	60%
Seed-150B-MoE	73%	79%

Table 4 Consistent Algorithm Rankings. Seed-150B- MoE results are ablation-only with limited steps.

7 Related Work

Test-time scaling [4, 18-20] such as OpenAI's o1 [1] and DeepSeek's R1 [2] have catalyzed a profound paradigm shift in LLMs [21, 22]. By enabling extended CoT reasoning [23] and eliciting sophisticated reasoning capabilities, these methods empower LLMs to excel in complex mathematical and coding tasks, including those from competitions like the AIME and Codeforces. At the core of this transformation is large-scale reinforcement learning, which facilitates the emergence of complex reasoning behaviors - such as self-verification and iterative refinement. However, the critical methodologies and algorithms underpinning scalable RL training have largely remained obscure, often omitted from the technical documentation of existing reasoning models [1, 2, 21-23]. In this paper, we introduce an SOTA-level model Seed-Thinking-v1.5 and introduce the details to achieve the performance from three aspects: Data, RL algorithm, and RL infrastructure.

8 Conclusion

We introduce a superb reasoning model named Seed-Thinking-v1.5, which achieves excellent performance across both reasoning tasks and non-reasoning tasks. It utilizes advanced RL techniques to improve the thinking ability stably and reliably by attaining 86.7% on AIME24, 74.0% on AIME25 and 55.0% on Codeforces In the future, we plan to investigate more efficient RL recipes and explore more

References

[1] OpenAI. 使用大型语言模型学习推理，2024。

[2] DeepSeek-AI. Deepseek-r1：通过强化学习激励大型语言模型的推理能力，2025。

[3] Google DeepMind. Gemini 2.5：我们最智能的人工智能模型，2025。

[4] Anthropic. Claude 3.7 十四行诗和 Claude 代码，2025。

[5] 岳宇、袁玉峰、余庆英、左晓辰、朱若飞、徐文远、陈家泽、王成毅、范天天、杜正银、魏象鹏、余象宇、刘高宏、刘俊才、刘灵军、林海斌、林志琦、马博磊、张驰、张莫凡、张旺、朱航、张如、刘欣、王明轩、吴永辉、颜琳。Vapo：高效可靠的高级推理任务强化学习，2025。

[6] 余庆英、张征、朱若飞、袁玉峰、左晓辰、岳宇、范天天、刘高宏、刘灵军、刘欣、林海斌、林志琦、马博磊、盛光明、童玉轩、张驰、张莫凡、张旺、朱航、朱金华、陈家泽、陈江杰、王成毅、余红丽、戴伟楠、宋玉轩、魏象鹏、周昊、刘静晶、马卫英、张雅琴、颜琳、乔木、吴永辉、王明轩。Dapo：大规模开源的大规模语言模型强化学习系统，2025。

[7] 字节跳动。抖宝-1.5-pro，2025。

[8] 沈伟、刘观麟、武征、朱若飞、杨青平、辛超、岳宇、颜琳。探索人类反馈强化学习中的数据扩展趋势和效果，2025。

[9] 徐文远、左晓辰、辛超、岳宇、颜琳、吴永辉。统一的配对框架用于 RLHF：连接生成奖励建模和策略优化，2025。

[10] 袁玉峰、岳宇、朱若飞、范天天、颜琳。PPO 在长链推理中的崩溃背后是什么？价值优化掌握着秘密。arXiv 预印本 arXiv:2503.01491, 2025。

[11] 盛光明、张驰、叶子凌锋、吴锡斌、张旺、张如、彭阳华、林海斌、吴川。Hybridflow：灵活高效的 RLHF 框架。arXiv 预印本 arXiv:2409.19256, 2024。

[12] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, William Paul, Michael I. Jordan, 和 Ion Stoica。Ray：新兴 AI 应用的分布式框架。CoRR, abs/1712.05889, 2017。

[13] Yao Zhewei, Yazdani Aminabadi Reza, Ruwase Olatunji, Rajbhandari Samyam, Wu Xiaoxia, Awan Ammar Ahmad, Rasley Jeff, Zhang Minjia, Li Conglong, Holmes Connor, Zhou Zhongzhu, Wyatt Michael, Smith Molly, Kurilenko Lev, Qin Heyang, Tanaka Masahiro, Che Shuai, Song Shuaiwen Leon, 和 He Yuxiong。Deepspeed-chat：简单、快速且经济实惠的 ChatGPT 类模型在所有规模下的 RLHF 训练，2023。

[14] Narendra Karmarkar 和 Richard M Karp。集合划分的差分方法。计算机科学部（EECS），加州大学伯克利分校，1982。

[15] 天琪陈、冰徐、赤渊张、卡洛斯 Guestrin。以次线性内存成本训练深度网络。arXiv 预印本 arXiv:1604.06174, 2016。[16] 郑连敏，李柱翰，张浩，庄永豪，陈志峰，黄艳萍，王一达，许远忠，卓丹阳，Eric P Xing 等。Alpa：为分布式深度学习自动化跨运算符和运算符内并行性。在第 16 届 USENIX 操作系统设计与实现研讨会（OSDI 22），第 559-578 页，2022 年。

[17] 万伯瑞，韩明基，盛屹尧，彭扬华，林海斌，张墨凡，赖智超，余梦涵，张俊达，宋祖全，刘欣，吴川。ByteCheckpoint：用于大型基础模型开发的统一检查点系统，2025 年。

[18] Qwen。QWQ-32B：拥抱强化学习的力量，2024 年。

[19] XAI。Grok 3 Beta - 推理代理的时代，2024 年。

[20] Google DeepMind。Gemini 2.0 闪电思维，2024 年。

[21] Tom Brown，Benjamin Mann，Nick Ryder，Melanie Subbiah，Jared D Kaplan，Prafulla Dhariwal，Arvind Neelakantan，Pranav Shyam，Girish Sastry，Amanda Askell 等。语言模型是少量样本学习者。神经信息处理系统进展，33:1877-1901，2020 年。

* 表示已离开团队的成员

challenging tasks with thinking mode to push the boundary of model's intelligence. Moreover, general reward modeling with comparable accuracy as verifier would also be a compelling research direction.

Contributions and Acknowledgments

Core Contributors

(Author names randomly shuffled)
Yufeng Yuan
Yu Yue
Mingxuan Wang
Xiaochen Zuo
Jiaze Chen
Lin Yan
Wenyuan Xu
Chi Zhang
Xin Liu
Chengyi Wang
TianTian Fan
Lingjun Liu
Qiying Yu
Xiangpeng Wei
Zhiqi Lin

Contributors

Ruofei Zhu
Qingping Yang
Chengzhi Wei
Jerry He
Guanlin Liu
Zheng Wu*
Xiangyu Yu
Zhicheng Liu
Jingjing Xu
Jiangjie Chen
Haojie Pan
Shengding Hu*
Zhengyin Du
Wenqi Wang
Zewei Sun
Chenwei Lou
Bole Ma
Zihan Wang
Mofan Zhang

[22] OpenAI。GPT4 技术报告。arXiv 预印本 arXiv:2303.08774，2023 年。

[23] Jason Wei，Xuezhi Wang，Dale Schuurmans，Maarten Bosma，Fei Xia，Ed Chi，Quoc V Le，Denny Zhou 等。链式思维提示法激发大型语言模型中的推理能力。神经信息处理系统进展，35:24824-24837，2022 年。

Appendix

A Case Study on Verifier

表 5 展示了对 Seed-Verifier 和 Seed-Thinking-Verifier 的案例研究。非常明显，Seed-Verifier 在处理答案复杂的问题时表现得非常吃力。相反，Seed-Thinking-Verifier 通过进行逐步分析，能够提供准确的判断结果。得益于其详细的思考过程，Seed-Thinking-Verifier 表现出显著的灵活性，并且可以有效地推广到几乎任何领域。

B Case Study on Creative Writing

在表 6、7、8 中，我们展示了中英文的例子，以展示我们模型在创意写作方面的能力。每个例子分为三个不同的部分：原始用户提示、模型的思维过程和模型的最终响应。

表 5 Seed-Verifier 和 Seed-Thinking-Verifier 的案例研究。

Table 6 Case 1 on Creative Writing.

Table 7 Case 2 on Creative Writing.

Wang Zhang
Gaohong Liu
Kaihua Jiang
Haibin Lin
Ru Zhang
Juncai Liu
Li Han
Jinxin Chi

Pretraining Efforts

Wenqiang Zhang Jiayi Xu Jun Yuan Zhen Xiao Yuqiao Xian Jingqiao Wu Kai Hua Na Zhou Jianhui Duan Heyang Lu Changbao Wang Jinxiang Ou Shihang Wang Xiaoran Jin Xuesong Yao Chengyin Xu Wenchang Ma Zhecheng An Renming Pang Xia Xiao Jing Su Yuyu Zhang Tao Sun Kaibo Liu Yifan Sun Kai Shen Sijun Zhang Yiyuan Ma Xingyan Bin Ji Li Yao Luo Deyi Liu Shiyi Zhan Yunshui Li Yuan Yang Defa Zhu Ke Shen Chenggang Li Xun Zhou Liang Xiang

Team Lead

Yonghui Wu

References

[1] OpenAI. Learning to reason with llms, 2024.

[2] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[3] Google DeepMind. Gemini 2.5: Our most intelligent ai model, 2025.

[4] Anthropic. Claude 3.7 sonnet and claude code, 2025.

[5] Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu and Lin Yan. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks, 2025.

[6] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025.

[7] ByteDance. Doubao-1.5-pro, 2025.

[8] Wei Shen, Guanlin Liu, Zheng Wu, Ruofei Zhu, Qingping Yang, Chao Xin, Yu Yue, and Lin Yan. Exploring data scaling trends and effects in reinforcement learning from human feedback, 2025.

* represents members who have departed from the team

[9] Wenyuan Xu, Xiaochen Zuo, Chao Xin, Yu Yue, Lin Yan, and Yonghui Wu. A unified pairwise framework for rlhf: Bridging generative reward modeling and policy optimization, 2025.

[10] Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. What's behind ppo's collapse in long-cot? value optimization holds the secret. arXiv preprint arXiv:2503.01491, 2025.

[11] Guangming Sheng, Chi Zhang, Zilingfeng 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.

[12] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A distributed framework for emerging AI applications. CoRR, abs/1712.05889, 2017.

[13] Zhewei Yao, Reza Yazdani Aminabadi, Olatunji Ruwase, Samyam Rajbhandari, Xiaoxia Wu, Ammar Ahmad Awan, Jeff Rasley, Minjia Zhang, Conglong Li, Connor Holmes, Zhongzhu Zhou, Michael Wyatt, Molly Smith, Lev Kurilenko, Heyang Qin, Masahiro Tanaka, Shuai Che, Shuaiwen Leon Song, and Yuxiong He. Deepspeed-chat: Easy, fast and affordable rlhf training of chatgpt-like models at all scales, 2023.

[14] Narendra Karmarkar and Richard M Karp. The differencing method of set partitioning. Computer Science Division (EECS), University of California Berkeley, 1982.

[15] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. arXiv preprint arXiv:1604.06174, 2016.

[16] Lianmin Zheng, Zhuohan Li, Hao Zhang, Yonghao Zhuang, Zhifeng Chen, Yanping Huang, Yida Wang, Yuanzhong Xu, Danyang Zhuo, Eric P Xing, et al. Alpa: Automating inter-and {Intra-Operator} parallelism for distributed deep learning. In 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22), pages 559-578, 2022.

[17] Borui Wan, Mingji Han, Yiyao Sheng, Yanghua Peng, Haibin Lin, Mofan Zhang, Zhichao Lai, Menghan Yu, Junda Zhang, Zuquan Song, Xin Liu, and Chuan Wu. Bytecheckpoint: A unified checkpointing system for large foundation model development, 2025.

[18] Qwen. Qwq-32b: Embracing the power of reinforcement learning, 2024.

[19] XAI. Grok 3 beta - the age of reasoning agents, 2024.

[20] Google DeepMind. Gemini 2.0 flash thinking, 2024.

[21] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in neural information processing systems, 33:1877-1901, 2020.

[22] OpenAI. GPT4 technical report. arXiv preprint arXiv:2303.08774, 2023.

[23] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems, 35:24824-24837, 2022.

Appendix

A Case Study on Verifier

Table 5 presents case study for both the Seed-Verifier and the Seed-Thinking-Verifier. It is clearly evident that the Seed-Verifier struggles significantly when dealing with samples that have complex answers. In contrast, the Seed-Thinking-Verifier is capable of providing accurate judgment results by conducting

a step-by-step analysis. Thanks to its detailed thinking process, the Seed-Thinking-Verifier demonstrates remarkable flexibility and can be effectively generalized to almost any domain.

B Case Study on Creative Writing

In Table 6, 7, 8, we showcase examples in both Chinese and English to demonstrate our model's proficiency in creative writing. Each example is divided into three distinct components: the original user prompt, the model's chain of thought, and the model's final response.

Table 5 Case study for both Seed-Verifier and Seed-Thinking-Verifier.

Table 6 Case 1 on Creative Writing.

Table 7 Case 2 on Creative Writing.