CS 957, System-2 AI
Test-Time Scaling

Mahdieh Soleymani

April 2025

Sharif University of Technology

Recap: Direct policy differentiation

$$\theta^\star = \arg\max_\theta J(\theta)$$

$$J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[r(\tau)]$$

$$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

Slide adapted from Sergey Levine

Recap: What did we just do?

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t}) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

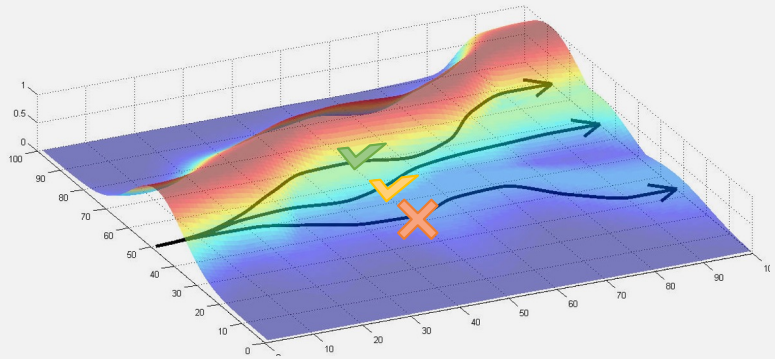
$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \underbrace{\nabla_\theta \log \pi_\theta(\tau_i)}_{\sum_{t=1}^{T} \nabla_\theta \log_\theta \pi_\theta(\mathbf{a}_{i,t}|\mathbf{s}_{i,t})} r(\tau_i)$$

maximum likelihood: $\nabla_\theta J_{\mathrm{ML}}(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \log \pi_\theta(\tau_i)$

good stuff is made more likely

bad stuff is made less likely

simply formalizes the notion of “trial and error”!



REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ (run the policy)
2. $\nabla_\theta J(\theta) \approx \sum_i \left(\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i) \right) \left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i) \right)$
3. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Slide adapted from Sergey Levine

Algorithm 1 "Vanilla" policy gradient algorithm

Initialize policy parameter θ, baseline b

for iteration=$1, 2, \ldots$ **do**

Collect a set of trajectories by executing the current policy

At each timestep in each trajectory, compute

the *return* $R_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}$, and

the *advantage estimate* $\hat{A}_t = R_t - b(s_t)$.

Re-fit the baseline, by minimizing $\|b(s_t) - R_t\|^2$,

summed over all trajectories and timesteps.

Update the policy, using a policy gradient estimate $\hat{g}$,

which is a sum of terms $\nabla_\theta \log \pi(a_t \mid s_t, \theta) \hat{A}_t$

end for

[Williams, 1992]

Importance sampling

$$\mathrm{E}_{x\sim p(x)}[f(x)] = \int f(x)p(x)dx$$

$$= \int f(x)\frac{p(x)}{q(x)}q(x)dx$$

$$= E_{x\sim q(x)}\left[f(x)\frac{p(x)}{q(x)}\right]$$

Surrogate Loss by Importance Sampling

$$J(\theta) = E_{\tau \sim \pi_{\theta_{old}}} \left[\frac{\pi_\theta(\tau)}{\pi_{\theta_{old}}(\tau)} r(\tau) \right]$$

$$\nabla J(\theta) = E_{\tau \sim \pi_{\theta_{old}}} \left[\frac{\nabla \pi_\theta(\tau)}{\pi_{\theta_{old}}(\tau)} r(\tau) \right]$$

$$\nabla J(\theta) \Big|_{\theta_{old}} = E_{\tau \sim \pi_{\theta_{old}}} \left[\frac{\nabla \pi_\theta(\tau)|_{\theta_{old}}}{\pi_{\theta_{old}}(\tau)} r(\tau) \right]$$

$$\nabla J(\theta) \Big|_{\theta_{old}} = E_{\tau \sim \pi_{\theta_{old}}} \left[\nabla \log \pi_\theta(\tau) \Big|_{\theta_{old}} r(\tau) \right]$$

Slide adapted from Pieter Abbeel

What's in a step-size?

- Step-sizing necessary as gradient is only first-order approximation
- Terrible step sizes, always an issue, but how about just not so great ones?
- Supervised learning
  - Step too far → next update will correct for it
- Reinforcement learning
  - Step too far → terrible policy
  - Next mini-batch: collected under this terrible policy!
  - Not clear how to recover short of going back and shrinking the step size

Slide adapted from Pieter Abbeel

TRPO

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t \right] \\ \text{subject to} \quad & \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \leq \delta. \end{aligned}$$

PPO v1

$$\max_\theta \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] - \beta \left(\hat{\mathbb{E}}_t \left[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)] \right] - \delta \right)$$

- Pseudocode:

for iteration=1, 2, . . . **do**

Run policy for T timesteps or N trajectories

Estimate advantage function at all timesteps

Do SGD on above objective for some number of epochs

Do dual descent update for beta

Slide adapted from Pieter Abbeel

Proximal Policy Optimization V2 – “Clipped Surrogate Loss”

$$L(s, a, \theta_k, \theta) = \min \left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s, a), \quad \text{clip} \left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1 - \epsilon, 1 + \epsilon \right) A^{\pi_{\theta_k}}(s, a) \right)$$

Slide adapted from Pieter Abbeel

Algorithm 1 PPO-Clip

Input: initial policy parameters θ_0, initial value function parameters ϕ_0

for $k = 0, 1, 2, \ldots$ **do**

Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.

Compute rewards-to-go $\hat{R}_t$.

Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}.

Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), \quad g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$$

typically via stochastic gradient ascent with Adam.

Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.

end for

Summary of RL algorithms

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t\left[\log \pi_\theta(a_t \mid s_t)\hat{A}_t\right]$$

$$L^{IS}_{\theta_{\text{old}}}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right]$$

$$L^{PPO}_{\theta_{old}}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}\hat{A}_t, \text{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right]$$

Problems of RL Approach

- Sample inefficiency
- Exploration-exploitation trade-off
  - E.g., data collecting by rejection sampling and then SFT has no exploitation strategy

RL Advantage

- Beyond human-curated Data
  - No dependency on human-generated data (constrained by data curated)

- Scalable learning algorithm
  - RL thrives by leveraging interaction and feedback loops.
  - Self-discovery: RL enables models to learn solutions through direct experience
  - Arbitrary Scaling: RL can scale effectively with increased compute

- The Bitter Lesson: RL aligns with the insights of this lesson
  - algorithms that leverage massive computation often outperform those relying heavily on human knowledge.

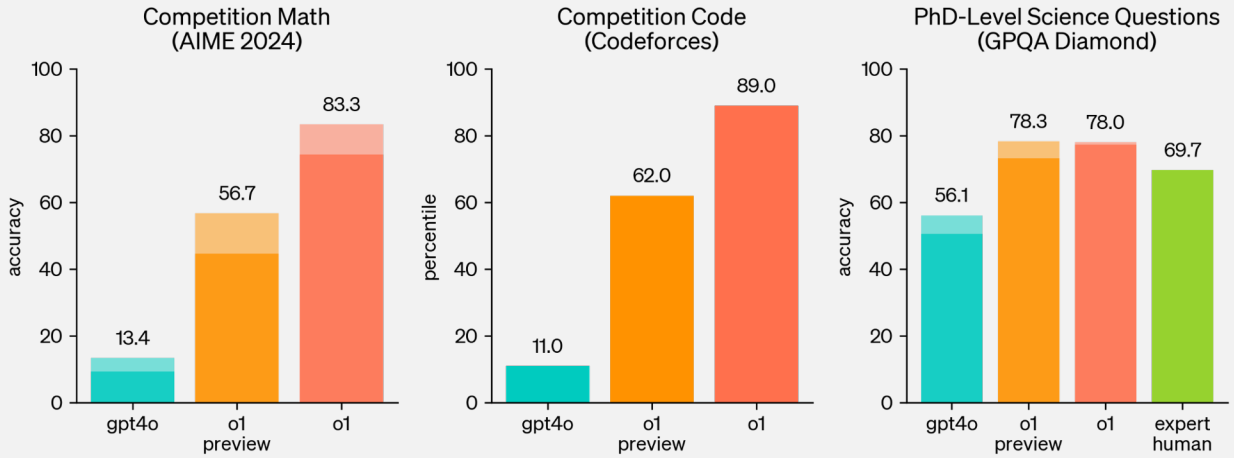
Large Reasoning Models (LRMs)

- language models trained via reinforcement learning to “reason” and “think through” extended chains of thought.

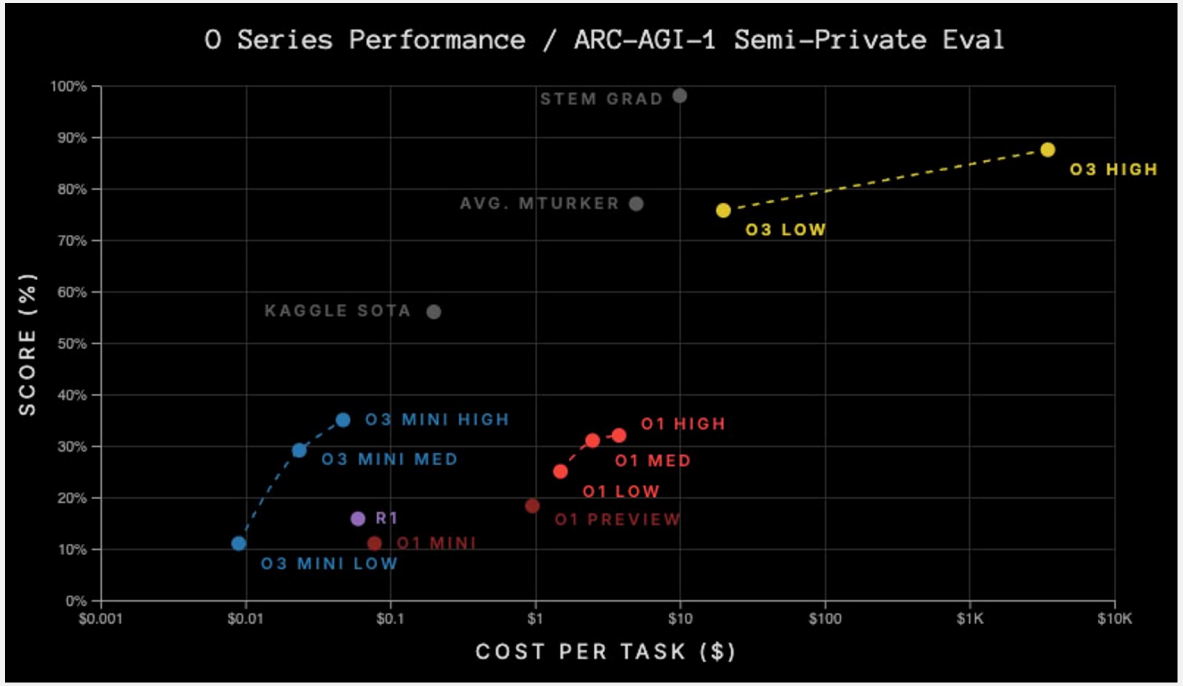
- emphasizing self-evolution rather than relying solely on supervised training data

Open AI LRMs

O1



O3



DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

DeepSeek-AI

research@deepseek.com

DeepSeek-R1-Zero and DeepSeek-R1

- DeepSeek-R1-Zero: trained via large-scale RL without SFT
  - remarkable reasoning capabilities
  - challenges such as poor readability, and language mixing.

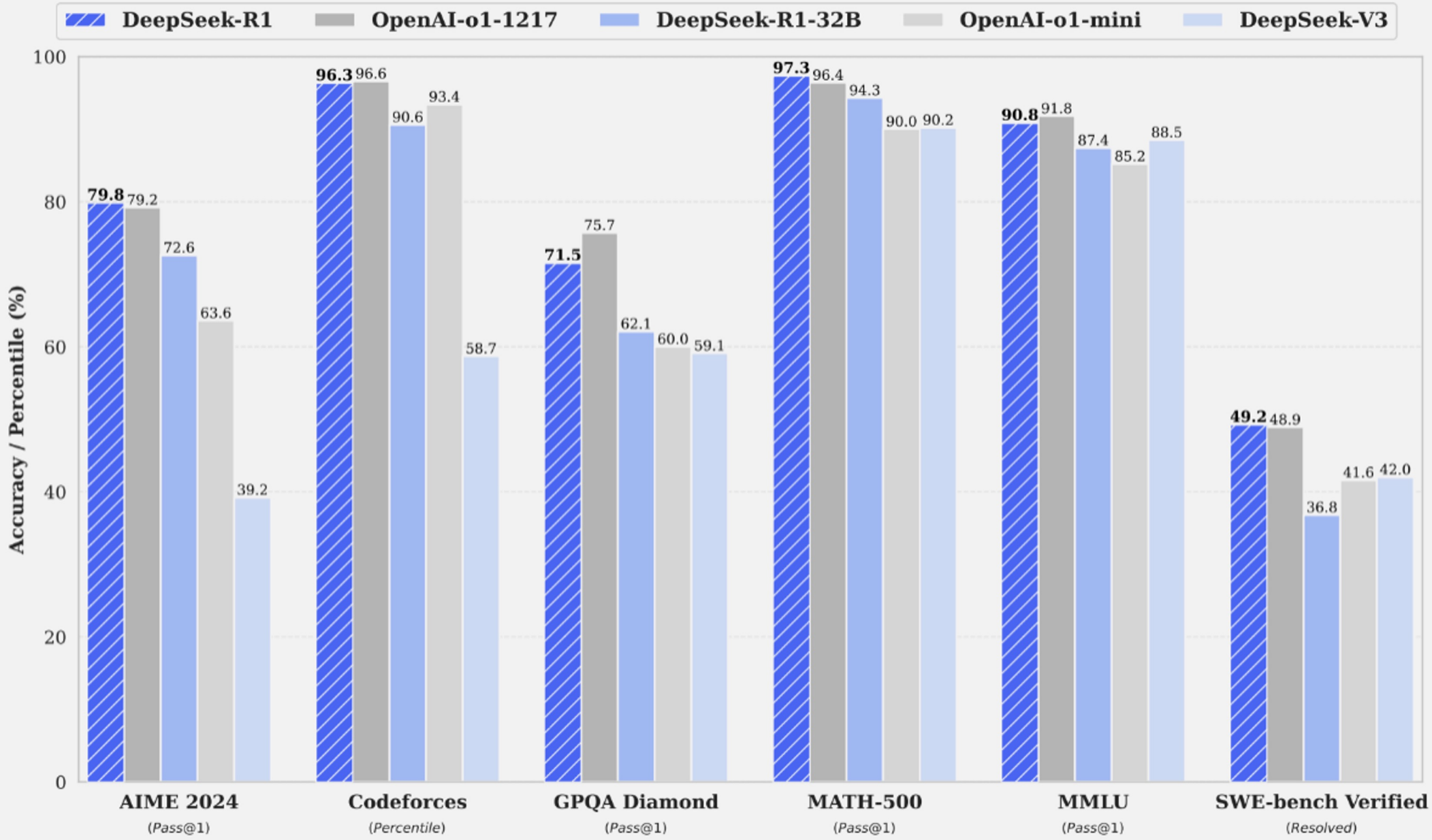
- DeepSeek-R1 incorporates multi-stage training pipeline

SFT and RL

- RL: more strategic thinking and self-improvement capabilities
  - it comes with challenges such as reward hacking, instability, and high computational costs.

- RL+SFT: more stable and generalizable improvements than pure RL.
  - Typically, a model is first trained with SFT on high-quality data and then further refined using RL to optimize specific behaviors.

- The exploratory benefits of RL with the stability of supervised learning is an interesting direction



DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, 2025

DeepSeek-R1-Zero and DeepSeek-R1

- DeepSeek-R1-Zero: trained via large-scale RL without SFT
  - remarkable reasoning capabilities
  - challenges such as poor readability, and language mixing.

- DeepSeek-R1 incorporates multi-stage training

DeepSeek-R1-Zero

- Self-evolution through a pure RL process
  - DeepSeek-V3-Base as the base model
  - GRPO as the RL framework

Reward Model

- Reward model usually suffers from the reward hacking problem
- Why is not it neccessay to have a trained reward model for RL approach as opposed to repeated sampling methods (with external verifier)?

Reward Model of DeepSeek-R1-Zero

- It does not apply the ORM or PRM in developing DeepSeek-R1-Zero
  - Reward model may suffer from reward hacking
  - Retraining the reward model needs additional training resources

- Rule-based reward model

$$R(\hat{y}, y) = \begin{cases} 1, & \texttt{is\_equivalent}(\hat{y}, y) \\ -1, & \text{otherwise} \end{cases}$$

Reward Model

- Accuracy rewards: The accuracy reward model evaluates whether the response is correct.
  - rule-based verification of correctness for math problems
  - for LeetCode problems, a compiler is used to generate feedback based on predefined test cases.

- Format rewards: enforces the model to put its thinking process between ‘<think>’ and ‘</think>’ tags.

Training Template

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think> <answer> answer here </answer>. User: prompt. Assistant:

Table 1 | Template for DeepSeek-R1-Zero. prompt will be replaced with the specific reasoning question during training.

Group Relative Policy Optimization (GRPO)

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$

$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right)$$

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{\text{std}(\{r_1, r_2, \cdots, r_G\})}$$

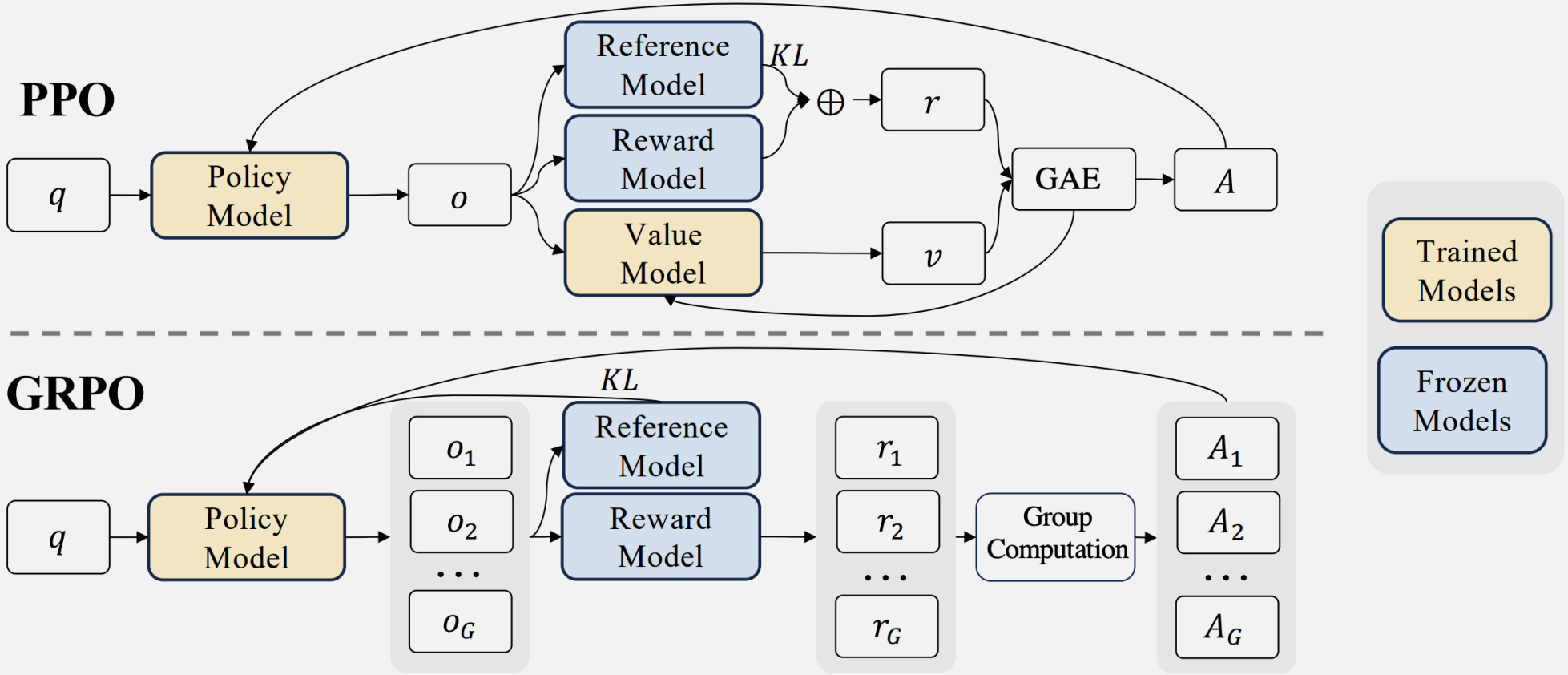
GRPO: Advantage Computation

- The advantage of the i-th response is calculated by normalizing the group-level rewards

$$A_i = \frac{r_i - \mathbf{mean}(\{r_1, r_2, \cdots, r_G\})}{\mathbf{std}(\{r_1, r_2, \cdots, r_G\})}$$

- GRPO computes the objective at the sample-level.
  - first calculates the mean loss within each generated sequence, before averaging the loss of different samples.

GRPO



Z. Shao et al., DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, 2024

q

"Solve for x: 2x + 5 = 11"

$o_1, \ldots, o_G \sim \pi_\theta$

Output 1:
2x = 11 - 5 → 2x = 6 → x = 3

Output 2:
2x = 11 + 5 = 16, so x = 8

Output 3:
x = (11 - 5) / 2 = 6 / 2 = 3

Output 4:
Let 2x = y. y + 5 = 11. y = 6. x = 3.

$r(o_1), \ldots, r(o_G)$

Output 1: Score: 93
2x = 11 - 5 → 2x = 6 → x = 3

Output 2: Score: 60
2x = 11 + 5 = 16, so x = 8

Output 3: Score: 69
x = (11 - 5) / 2 = 6 / 2 = 3

Output 4: Score: 25
Let 2x = y. y + 5 = 11. y = 6. x = 3.

$\bar{r} = 61.8$

$A_1, \ldots, AG$

Output 1: Adv: 31.3 👍 Positive
2x = 11 - 5 → 2x = 6 → x = 3

Output 2: Adv: -1.8 👎 Negative
2x = 11 + 5 = 16, so x = 8

Output 3: Adv: 7.3 👍 Positive
x = (11 - 5) / 2 = 6 / 2 = 3

Output 4: Adv: -36.8 👎 Negat
Let 2x = y. y + 5 = 11. y = 6. x = 3.

[GRPO Demo by Gemini]

DeepSeek-R1-Zero

- Self-evolution through a pure RL process
  - DeepSeek-V3-Base as the base model
  - GRPO as the RL framework

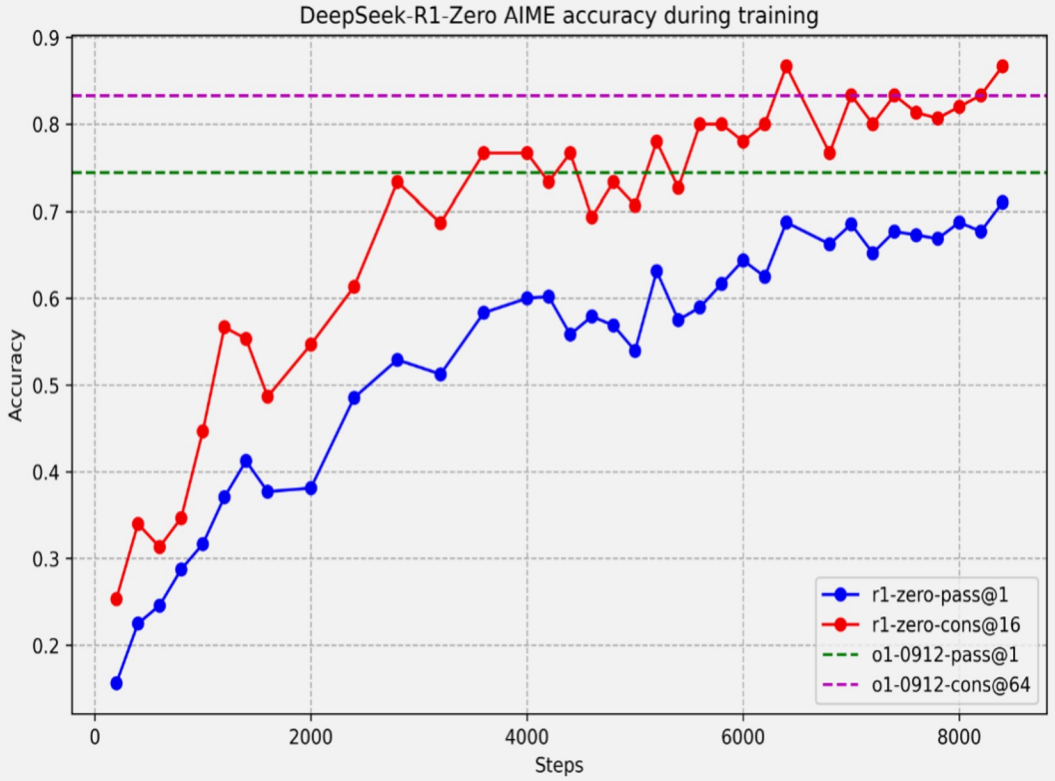
- After thousands of RL steps, super performance on reasoning
  - pass@1 score on AIME 2024 increases from 15.6% to 71.0%
  - with majority voting, improves to 86.7%

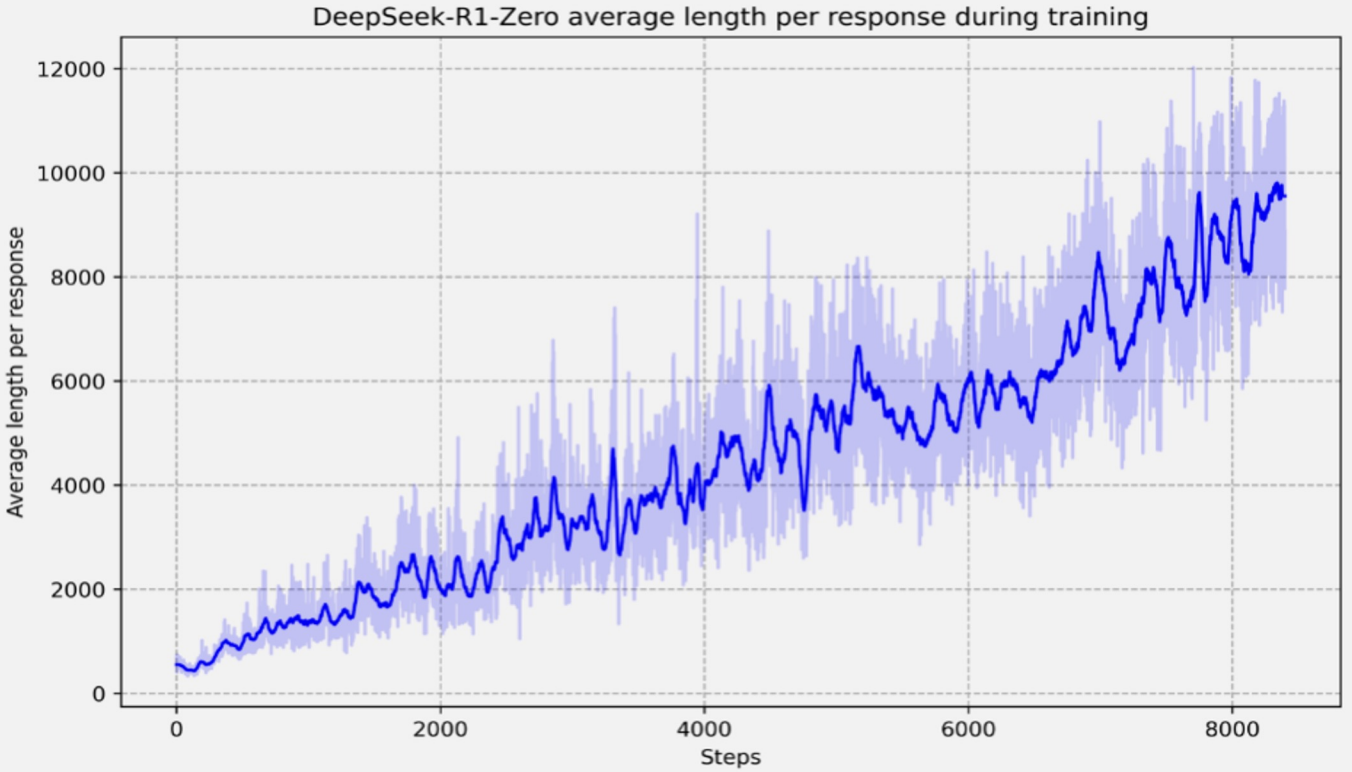
- Challenges such as poor readability, and language mixing

Results on reasoning-related benchmarks

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCode Bench	CodeForces
	pass@1	cons@64	pass@1	pass@1	pass@1	rating
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	1820
OpenAI-o1-0912	74.4	83.3	94.8	77.3	63.4	1843
DeepSeek-R1-Zero	71.0	86.7	95.9	73.3	50.0	1444

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, 2025





DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, 2025

Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a+x}} = x$ is equal to

Response: <think>

To solve the equation $\sqrt{a - \sqrt{a+x}} = x$, let's start by squaring both $\cdots$

$$\left(\sqrt{a - \sqrt{a+x}}\right)^2 = x^2 \implies a - \sqrt{a+x} = x^2.$$

Rearrange to isolate the inner square root term:

$$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$$

$\cdots$

Wait, wait. Wait. That's an aha moment I can flag here.

Let's reevaluate this step-by-step to identify if the correct sum can be $\cdots$

We started with the equation:

$$\sqrt{a - \sqrt{a+x}} = x$$

First, let's square both sides:

$$a - \sqrt{a+x} = x^2 \implies \sqrt{a+x} = a - x^2$$

Next, I could square both sides again, treating the equation: $\cdots$

$\cdots$

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, 2025

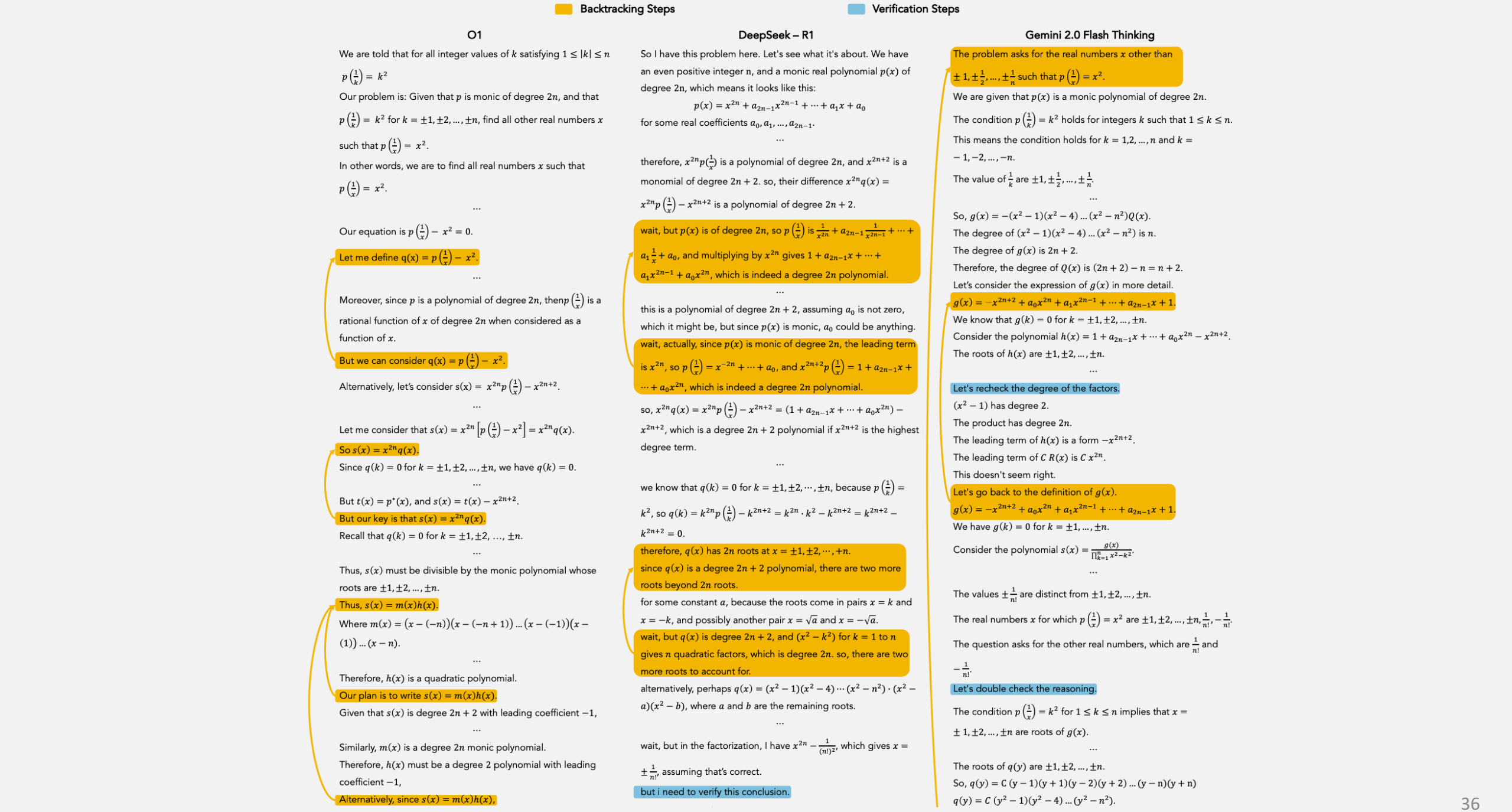
RL Magic

Hundreds to thousands of reasoning tokens, allowing the model to explore and refine its thought processes

Reflection (revisiting and reevaluating previous steps) and the exploration of alternative approaches arise spontaneously.

Self-evolution by RL

- **Intrinsic development** within the model instead of external adjustment
- RL allows the model to explore and refine its thought processes
- The **self-evolution** shows how RL can drive a model to improve its reasoning capabilities autonomously.
  - **reflection**—where the model revisits and reevaluates its previous steps
  - **exploration** of alternative approaches to problem-solving



Xiang et al., Towards System 2 Reasoning in LLMs: Learning How to Think With Meta Chain-of-Thought, 2025

DeepSeek-R1-Zero: Summary

Self-verification, reflection, and generating long CoTs capabilities

Reasoning capabilities purely through RL

DeepSeek-R1-Zero: Summary

Self-verification, reflection, and generating long CoTs capabilities

Reasoning capabilities purely through RL

Drawback of DeepSeek-R1-Zero: poor readability, and language mixing

DeepSeek-R1

- DeepSeek-R1 has a multi-stage training pipeline:
  - **Cold-start SFT**
  - **Reasoning-oriented RL**
  - **SFT** for retaining reasoning and non-reasoning capabilities
  - **RL for alignment**

DeepSeek-R1

- DeepSeek-R1 has a multi-stage training pipeline:
  - **Cold-start SFT**: collecting thousands of human-friendly data to fine-tune DeepSeek-V3-Base
  - **Reasoning-oriented RL** like DeepSeek-R1-Zero
  - **SFT:** rejection sampling on the RL checkpoint combined with supervised data from DeepSeek-V3 in other domains and then retrain the DeepSeek-V3-Base
  - **RL process**, prompts from all scenarios to align with human preferences

Cold Start SFT

Gathering **long CoT** by prompting

Refining them by **human annotators**

Fine-tune the based model as the initial model for RL

Cold Start SFT

- Gathering **long CoT** by prompting
  - **few-shot prompting** with a **long CoT** as an example
  - **prompting** to generate detailed answers with reflection and verification
  - gathering **DeepSeek-R1-Zero outputs** in a readable format, and refining them by human annotators.
- Fine-tune the base model to provide the initial model for RL

Reasoning-oriented RL

Similar to DeepSeek-R1-Zero

Rule-based accuracy rewards like DeepSeek-R1-Zero is used

Language consistency reward along accuracy reward during RL training to mitigate language mixing

Rejection Sampling and SFT

- Reasoning data: reasoning trajectories by performing rejection sampling on trajectories from the RL checkpoint
  - For each prompt, multiple responses are sampled and retained only the correct ones.
    - Verifier: feeding ground-truth and model predictions to DeepSeek-V3 for judgment.
  - ~ 600k reasoning related training samples are collected

- Non-Reasoning data:
  - reuse portions of the SFT dataset of DeepSeek-V3 as data of general-purpose tasks.
  - For certain non-reasoning tasks, DeepSeek-V3 is used to generate a potential CoT before answering the question by prompting.
  - ~200k training samples that are unrelated to reasoning

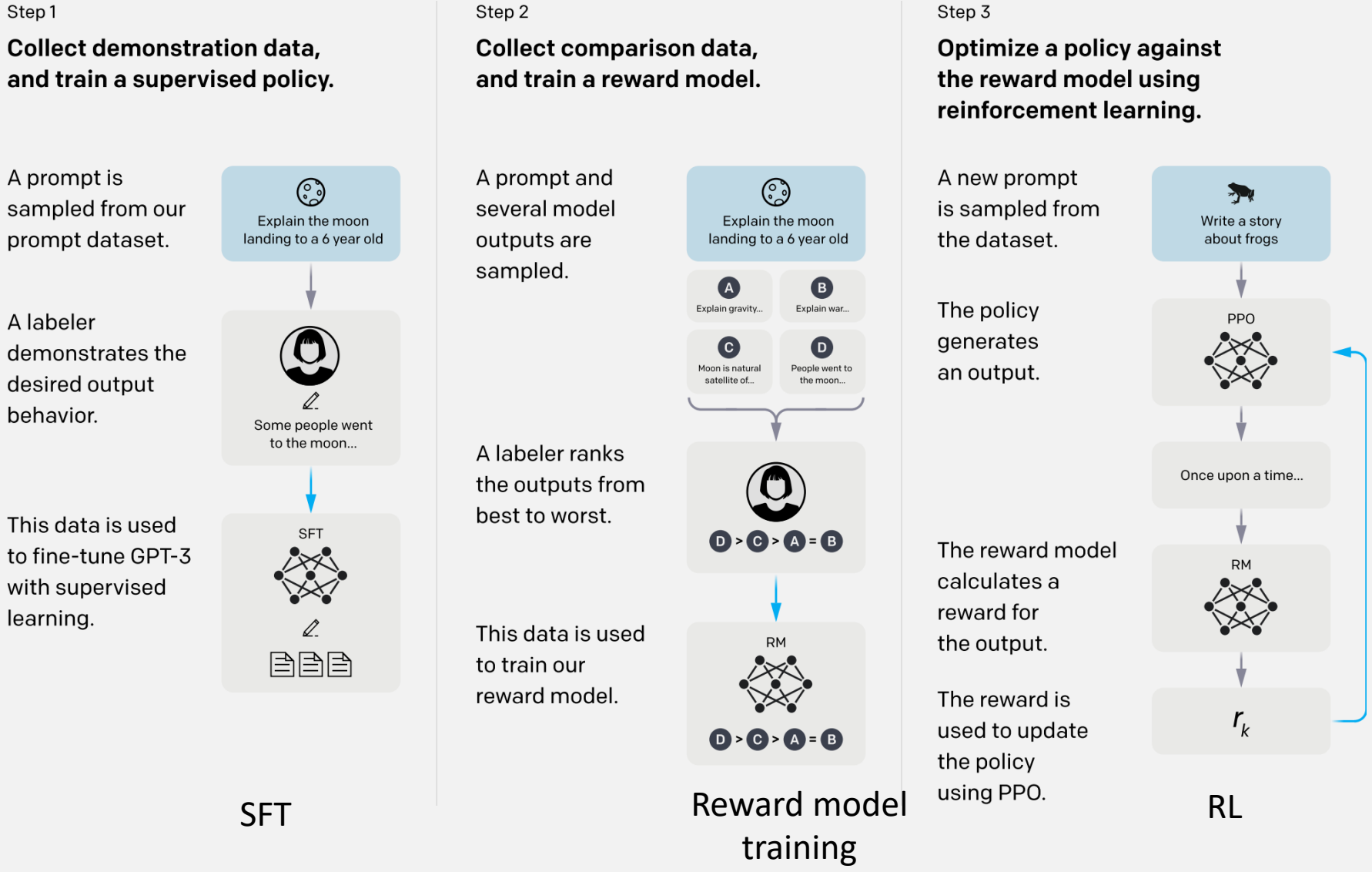
RL for all Scenarios

RL to further align the model with human preferences

Improving helpfulness and harmlessness while simultaneously refining its reasoning capabilities.

The model is trained using a combination of reward signals and diverse prompt distributions.

Instrucion Tuning and RLHF



Quyang et al., Training language models to follow instructions with human feedback, 2022

How do we model human preferences?

- For a question, ask for **pairwise comparisons**, which can be more reliable

An earthquake hit San Francisco. There was minor property damage, but no injuries. > A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage. > The Bay Area has good weather but is prone to earthquakes and wildfires.

RLHF: Pairwise preferences

- Based on DeepSeek-V3 pipeline and adopt a similar distribution of training prompts.

- Assessment by users
  - Helpfulness
  - Harmlessness
  - Honestness

How do we model human preferences?

An earthquake hit San Francisco. There was minor property damage, but no injuries. > A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage. > The Bay Area has good weather but is prone to earthquakes and wildfires.

- For each LM sample s, we have a reward model $RM_\phi(s)$ that predicts human reward

$$J_{RM}(\phi) = -\mathbb{E}_{(s^w, s^l)\sim D}\left[\log \sigma(RM_\phi(s^w) - RM_\phi(s^l))\right]$$

"winning" sample

"losing" sample

s^w should score higher than s^l

Combination of rewards

- For general data, reward models are used to capture human preferences in complex and nuanced scenarios.

- For reasoning data, rule-based rewards is utilized: math, code, and logical reasoning domains.

- The model is trained using a combination of reward signals

Distillation from DeepSeek-R1 to smaller dense models

Reasoning patterns of larger models can also help smaller models

Open-source models like Qwen and Llama are finetuned using 800k samples curated with DeepSeek-R1

Direct distillation from DeepSeek-R1 outperforms applying RL on them.

	Benchmark (Metric)	Claude-3.5-Sonnet-1022	GPT-4o 0513	DeepSeek V3	OpenAI o1-mini	OpenAI o1-1217	DeepSeek R1
	Architecture	-	-	MoE	-	-	MoE
	# Activated Params	-	-	37B	-	-	37B
	# Total Params	-	-	671B	-	-	671B
English	MMLU (Pass@1)	88.3	87.2	88.5	85.2	**91.8**	90.8
	MMLU-Redux (EM)	88.9	88.0	89.1	86.7	-	**92.9**
	MMLU-Pro (EM)	78.0	72.6	75.9	80.3	-	**84.0**
	DROP (3-shot F1)	88.3	83.7	91.6	83.9	90.2	**92.2**
	IF-Eval (Prompt Strict)	**86.5**	84.3	86.1	84.8	-	83.3
	GPQA Diamond (Pass@1)	65.0	49.9	59.1	60.0	**75.7**	71.5
	SimpleQA (Correct)	28.4	38.2	24.9	7.0	**47.0**	30.1
	FRAMES (Acc.)	72.5	80.5	73.3	76.9	-	**82.5**
	AlpacaEval2.0 (LC-winrate)	52.0	51.1	70.0	57.8	-	**87.6**
	ArenaHard (GPT-4-1106)	85.2	80.4	85.5	92.0	-	**92.3**
Code	LiveCodeBench (Pass@1-COT)	38.9	32.9	36.2	53.8	63.4	**65.9**
	Codeforces (Percentile)	20.3	23.6	58.7	93.4	**96.6**	96.3
	Codeforces (Rating)	717	759	1134	1820	**2061**	2029
	SWE Verified (Resolved)	**50.8**	38.8	42.0	41.6	48.9	49.2
	Aider-Polyglot (Acc.)	45.3	16.0	49.6	32.9	**61.7**	53.3
Math	AIME 2024 (Pass@1)	16.0	9.3	39.2	63.6	79.2	**79.8**
	MATH-500 (Pass@1)	78.3	74.6	90.2	90.0	96.4	**97.3**
	CNMO 2024 (Pass@1)	13.1	10.8	43.2	67.6	-	**78.8**
Chinese	CLUEWSC (EM)	85.4	87.9	90.9	89.9	-	**92.8**
	C-Eval (EM)	76.7	76.0	86.5	68.9	-	**91.8**
	C-SimpleQA (Correct)	55.4	58.7	**68.0**	40.3	-	63.7

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, 2025

DAPO: An Open-Source LLM Reinforcement Learning System at Scale

[1]ByteDance Seed [2]Institute for AI Industry Research (AIR), Tsinghua University
[3]The University of Hong Kong
[4]SIA-Lab of Tsinghua AIR and ByteDance Seed

Full author list in Contributions

Decoupled Clip and Dynamic s**A**mpling **P**olicy **O**ptimization (DAPO)

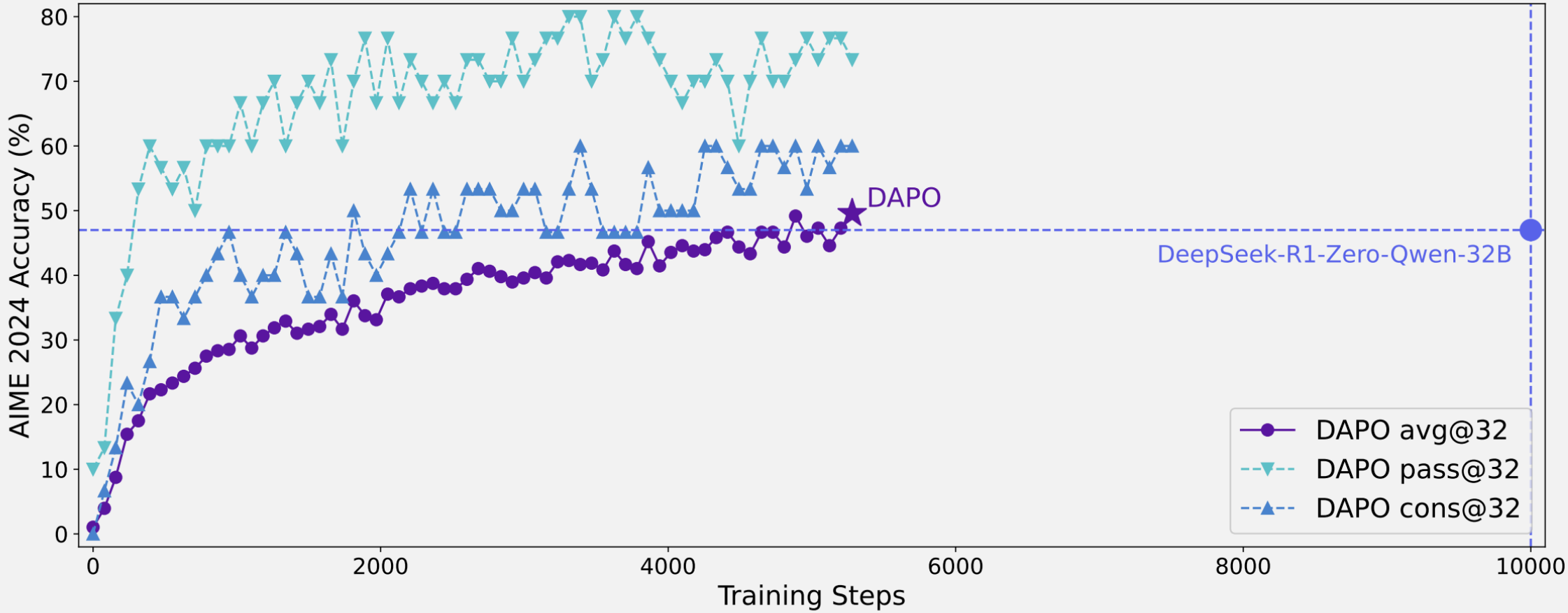
- For a group of outputs $\{o_i\}_{i=1}^{G}$ on each question q paired with the answer a, the objective is defined as:

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

$$\text{s.t.} \quad 0 < \left| \{o_i \mid \texttt{is\_equivalent}(a, o_i)\} \right| < G,$$

$$r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}, \quad \hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^{G})}{\text{std}(\{R_i\}_{i=1}^{G})}.$$

DAPO: Results on AIME 2024



AIME 2024 scores of DAPO on the Qwen2.5-32B base model, outperforming the previous SoTA DeepSeekR1-Zero-Qwen-32B using 50% training steps.

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

DAPO: Differences from GRPO

- **Clip-Higher** promotes the diversity of the system and avoids entropy collapse
- **Dynamic Sampling** improves training efficiency and stability
- **Token-Level Policy Gradient Loss** is critical in long-CoT RL scenarios
- **Overlong Reward Shaping** reduces reward noise and stabilizes training

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Clip-Higher

- Upper clipping threshold restricts the probability increase of low-probability tokens (constraining the diversity of the system)
- Example: $\varepsilon = 0.2$, $\pi_{\theta_{old}}(o_1|q) = 0.01$ and $\pi_{\theta_{old}}(o_2|q) = 0.9$.
  - $\pi_\theta(o_1|q)$ are bounded by 0.012 and $\pi_\theta(o_1|q)$ by 1.08
  - Thus, for tokens with a higher probability (e.g., 0.9) it is less constrained.

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Clip-Higher

- Upper clipping threshold restricts the probability increase of low-probability tokens (constraining the diversity of the system)
- Example: $\varepsilon = 0.2$, $\pi_{\theta_{old}}(o_1|q) = 0.01$ and $\pi_{\theta_{old}}(o_2|q) = 0.9$.
  - $\pi_\theta(o_1|q)$ are bounded by 0.012 and $\pi_\theta(o_1|q)$ by 1.08
  - Thus, for tokens with a higher probability (e.g., 0.9) it is less constrained.
- The lower and higher clipping range as ε_{low} and $\varepsilon_{\text{high}}$ are decoupled

$$\mathcal{J}_{\text{DAPO}}(\theta) = \mathbb{E}_{(q,a)\sim\mathcal{D},\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)}$$

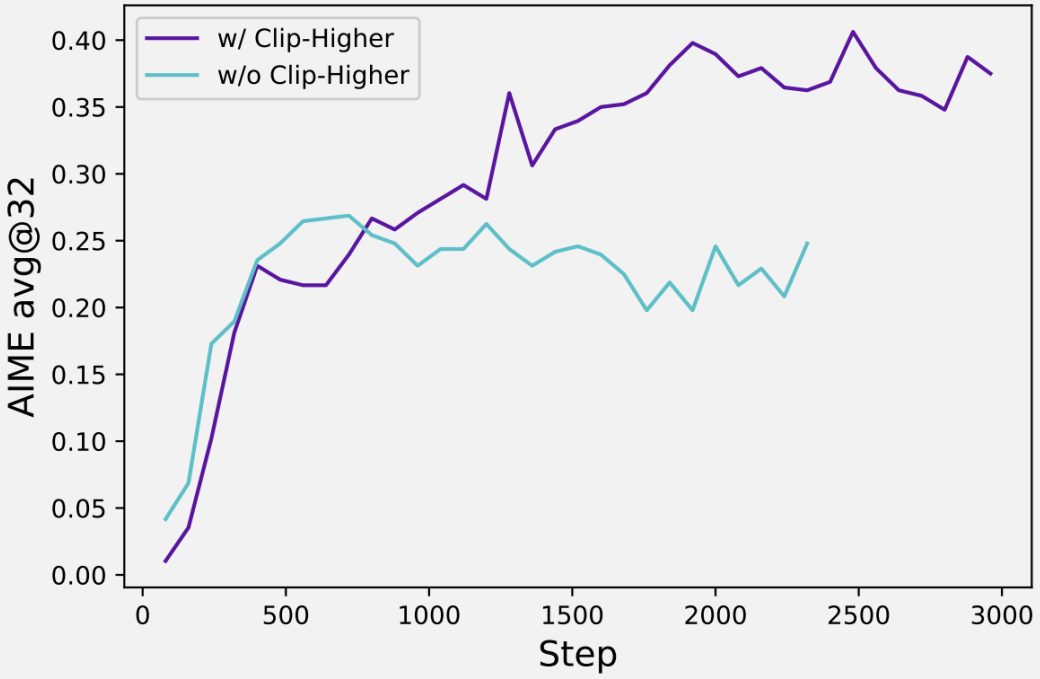
increase the value of $\varepsilon_{\text{high}}$

$$\left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\left(r_{i,t}(\theta)\hat{A}_{i,t},\ \text{clip}\left(r_{i,t}(\theta), 1-\varepsilon_{\text{low}}, 1+\varepsilon_{\text{high}}\right)\hat{A}_{i,t}\right)\right]$$

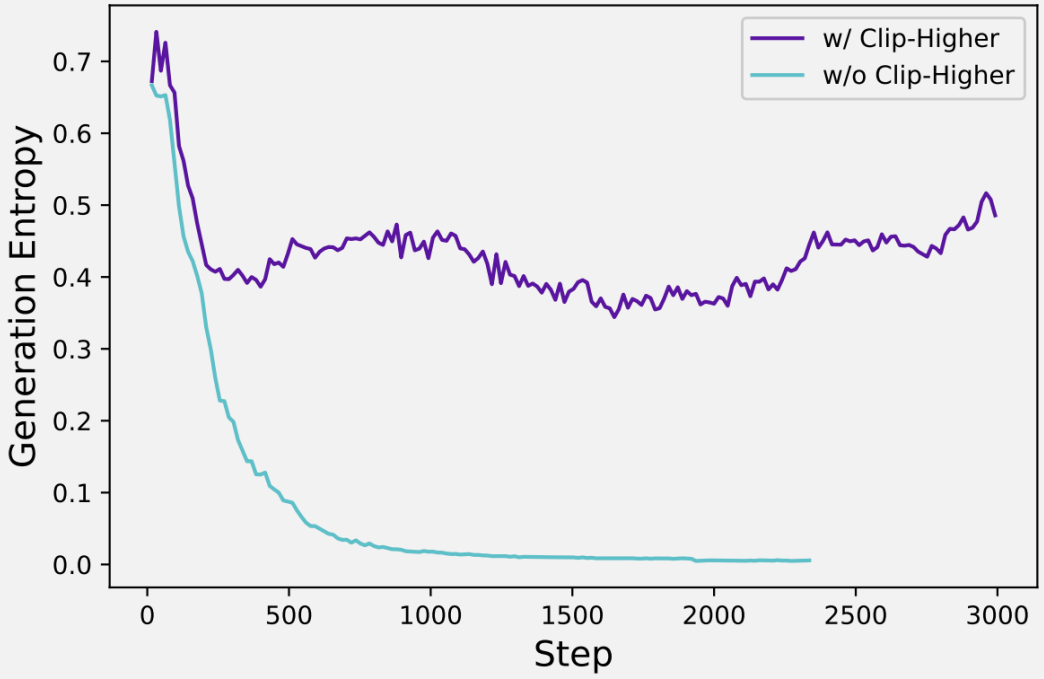
$$\text{s.t.}\quad 0 < \left|\{o_i \mid \texttt{is\_equivalent}(a, o_i)\}\right| < G.$$

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Clip-Higher (Results)



(a) Accuracies on AIME.



(b) Entropy of actor model.

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Dynamic Sampling

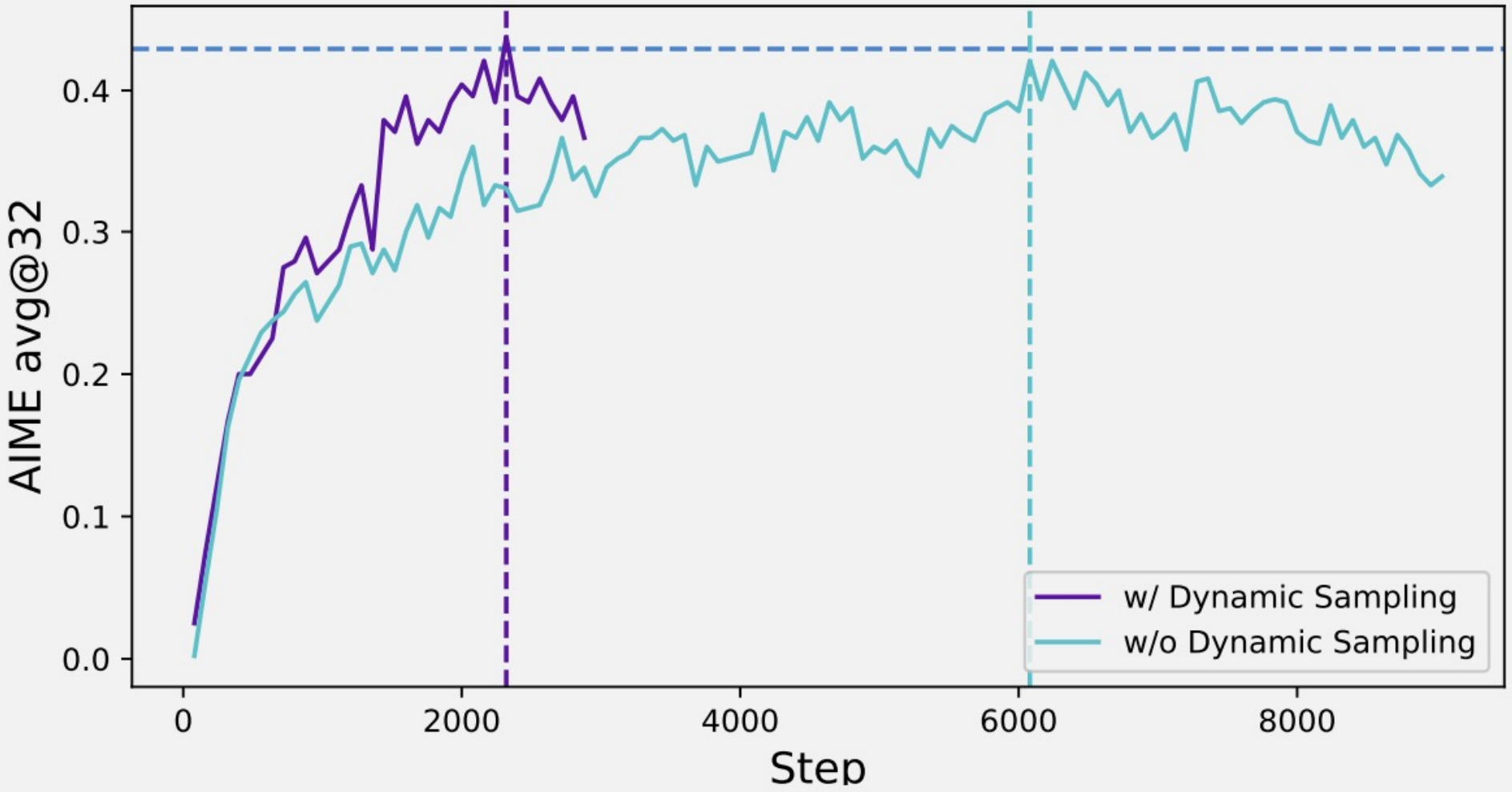
- The gradient-decreasing problem when some prompts have accuracy equal to 1 or 0.
  - If all outputs $\{o_i\}_{i=1}^{G}$ of a prompt receive the same reward, advantage for this group is 0.
  - A zero advantage results in no gradients for policy updates, thereby reducing sample efficiency
- Over-sample and filter out prompts with the accuracy equal to 1 and 0
- Before training, we keep sampling until the batch is fully filled with samples whose accuracy is neither 0 nor 1.

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

$$\text{s.t.} \quad 0 < \left| \{o_i \mid \texttt{is\_equivalent}(a, o_i)\} \right| < G.$$

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Dynamic Sampling (Results)



DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Token-Level Policy Gradient Loss

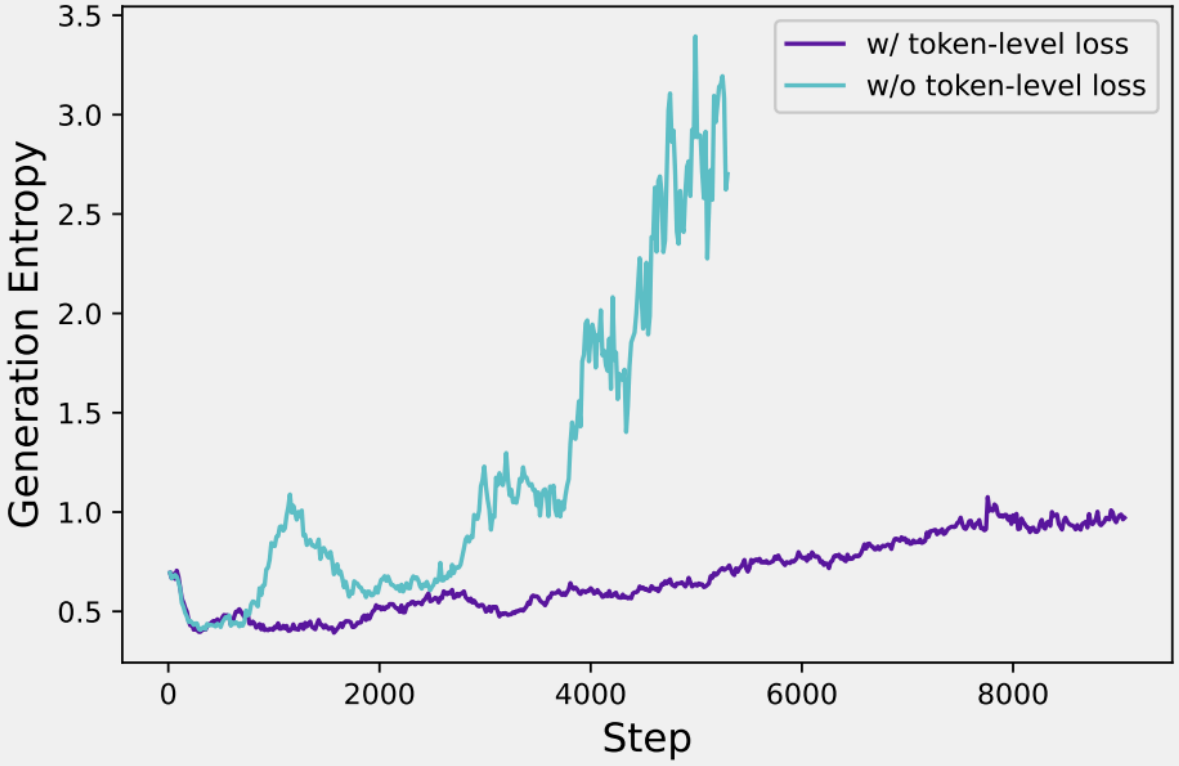
- Due to sample-level loss, all samples are assigned the same weight
- Tokens within longer responses may have a disproportionately lower contribution:
  - For high-quality long samples, it can impede the ability to learn reasoning-relevant patterns
  - Excessively long samples often exhibit low-quality patterns like gibberish and repetitive words.
    - Without effectively penalizing those patterns, an unhealthy increase in entropy and response length

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

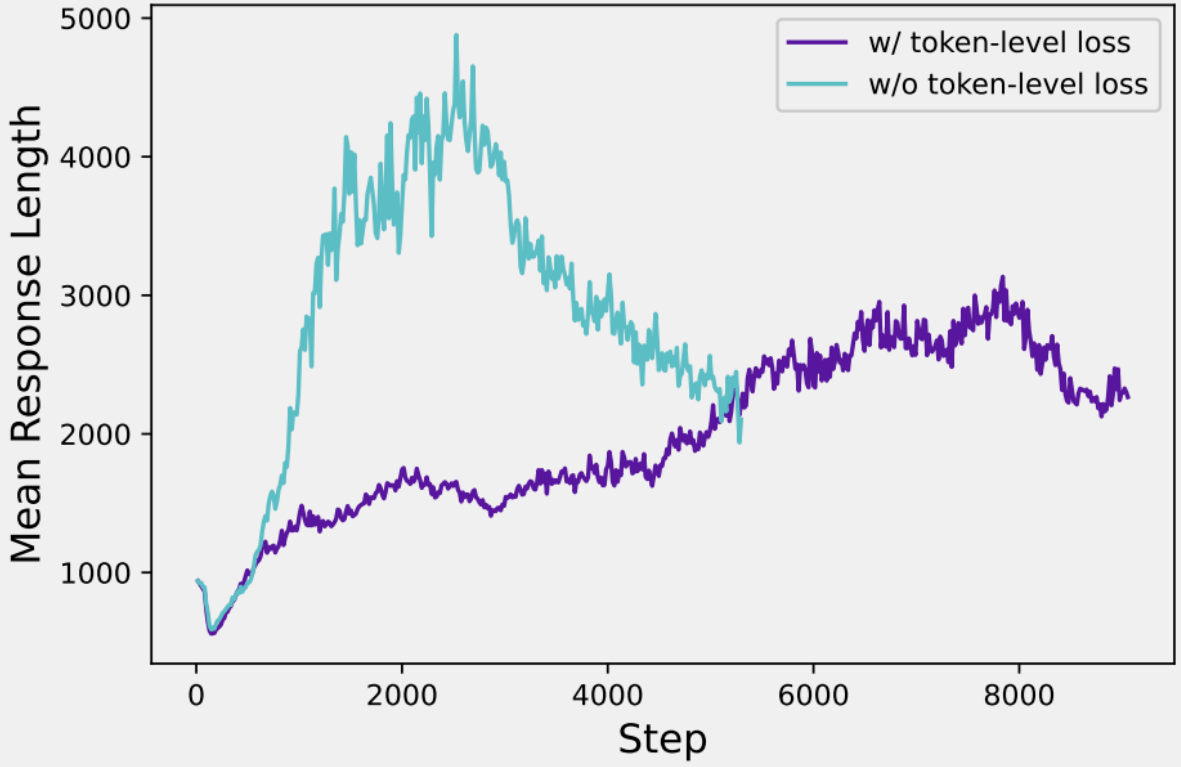
$$\text{s.t.} \quad 0 < \left| \{o_i \mid \texttt{is\_equivalent}(a, o_i)\} \right| < G.$$

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Token-Level Policy Gradient Loss (Results)



(a) Entropy of actor model's generation probabilities.



(b) Average length of actor model-generated responses

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Overlong Filtering

- In RL training, a maximum length is typically set for generation, with overlong samples truncated accordingly.

- By default, a punitive reward is assigned to truncated samples.
  - It may introduce noise into the training process, as a sound reasoning process can be penalized solely due to its excessive length.
  - Such penalties can potentially confuse the model regarding the validity of its reasoning process.

- Overlong Filtering strategy that masks the loss of truncated samples can be applied.

Soft Overlong Punishment

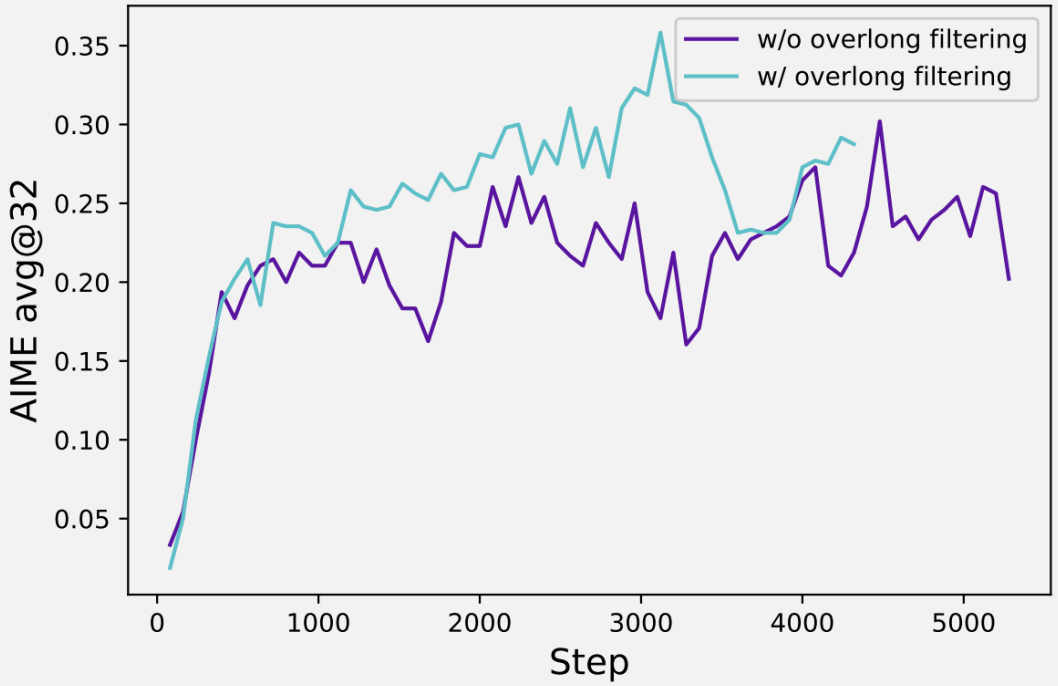
- A length-aware penalty designed to shape the reward for truncated samples.
  - When the response length exceeds the predefined len, a punishment interval is defined.
  - Within this interval, the longer the response, the greater the punishment it receives.

- This penalty is added to the original rule-based correctness reward

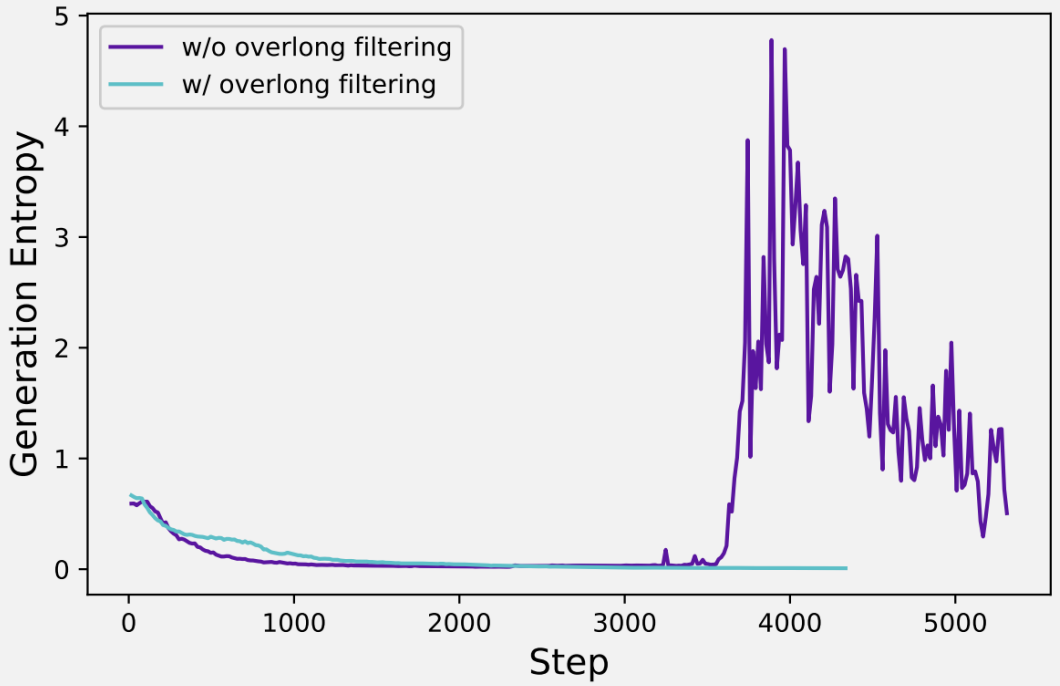
$$R_{\text{length}}(y) = \begin{cases} 0, & |y| \leq L_{\max} - L_{\text{cache}} \\ \frac{(L_{\max} - L_{\text{cache}}) - |y|}{L_{\text{cache}}}, & L_{\max} - L_{\text{cache}} < |y| \leq L_{\max} \\ -1, & L_{\max} < |y| \end{cases}$$

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Soft Overlong Punishment (Results)



(a) Performance on AIME.



(b) Entropy of actor model.

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

Algorithm 1 **DAPO**: **D**ecoupled Clip and **D**ynamic s**A**mpling **P**olicy **O**ptimization

Input initial policy model π_θ; reawrd model R; task prompts $\mathcal{D}$; hyperparameters $\varepsilon_{\texttt{low}}, \varepsilon_{\texttt{high}}$

for step = 1,...,M **do**
  Sample a batch $\mathcal{D}_b$ from $\mathcal{D}$
  Update the old policy model $\pi_{\theta_{old}} \leftarrow \pi_\theta$
  Sample G outputs $\{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot|q)$ for each question $q \in \mathcal{D}_b$
  Compute rewards $\{r_i\}_{i=1}^{G}$ for each sampled output o_i by running R
  Filter out o_i and add the remaining to the dynamic sampling buffer (**Dynamic Sampling** Equation (11))
  if buffer size $n_b < N$:
    continue
  For each o_i in the buffer, compute $\hat{A}_{i,t}$ for the t-th token of o_i (Equation (9))
  for iteration = 1, ..., μ **do**
    Update the policy model π_θ by maximizing the DAPO objective (Equation (8))

Output π_θ

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025.

DAPO: Ablation

Model	AIME24$_{\text{avg@32}}$
DeepSeek-R1-Zero-Qwen-32B	47
Naive GRPO	30
+ Overlong Filtering	36
+ Clip-Higher	38
+ Soft Overlong Punishment	41
+ Token-level Loss	42
+ Dynamic Sampling (**DAPO**)	**50**

DAPO: An Open-Source LLM Reinforcement Learning System at Scale, 2025

2025-4-1

Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model

Jingcheng Hu[1,2*], Yinmin Zhang[1], Qi Han[1], Daxin Jiang[1], Xiangyu Zhang[1], Heung-Yeung Shum[2]

[1]**StepFun,** [2]**Tsinghua University**

GitHub: `https://github.com/Open-Reasoner-Zero/Open-Reasoner-Zero`,
HuggingFace: `https://huggingface.co/Open-Reasoner-Zero`.

Open-Reasoner-Zero (ORZ)

- ORZ’s stable scaling resonates well with the bitter lesson
  - The most significant improvements stem from the scale of training data, model size, and training iterations
  - rather than the complexity of design choices.

- The most critical thing is how to design a simple and effective RL algorithm to scale up the training process.

ORZ: Key findings

- RL algorithm
  - **vanilla PPO** provides a remarkably stable and robust training process
  - GAE parameters play a critical role in PPO for reasoning tasks.
    - Specifically, setting λ = 1.0 and γ = 1.0, achieves the ideal balance for scale-up RL training.
- Minimal reward function design
  - simple rule-based reward function is not only sufficient but optimal
- Loss function
  - Stable training is achieved without relying on any KL-based regularization techniques (e.g., KL shaped rewards and loss)
- Scale up training data
  - Scaling up data quantity and diversity is crucial for Reasoner-Zero training.

ORZ

$$\mathcal{J}_{\text{PPO}}(\theta) = \mathbb{E}_{t,s_t,a_t\sim\pi_{\theta_{\text{old}}}}\left[\min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t, \text{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right],$$

$$\mathcal{J}_{\text{value}}(\phi) = \frac{1}{2}\mathbb{E}_{t,s_t,a_t\sim\pi_{\theta_{\text{old}}}}\left[\left(V_\phi(s_t) - R_t\right)^2\right],$$

Removal of KL divergence penalty

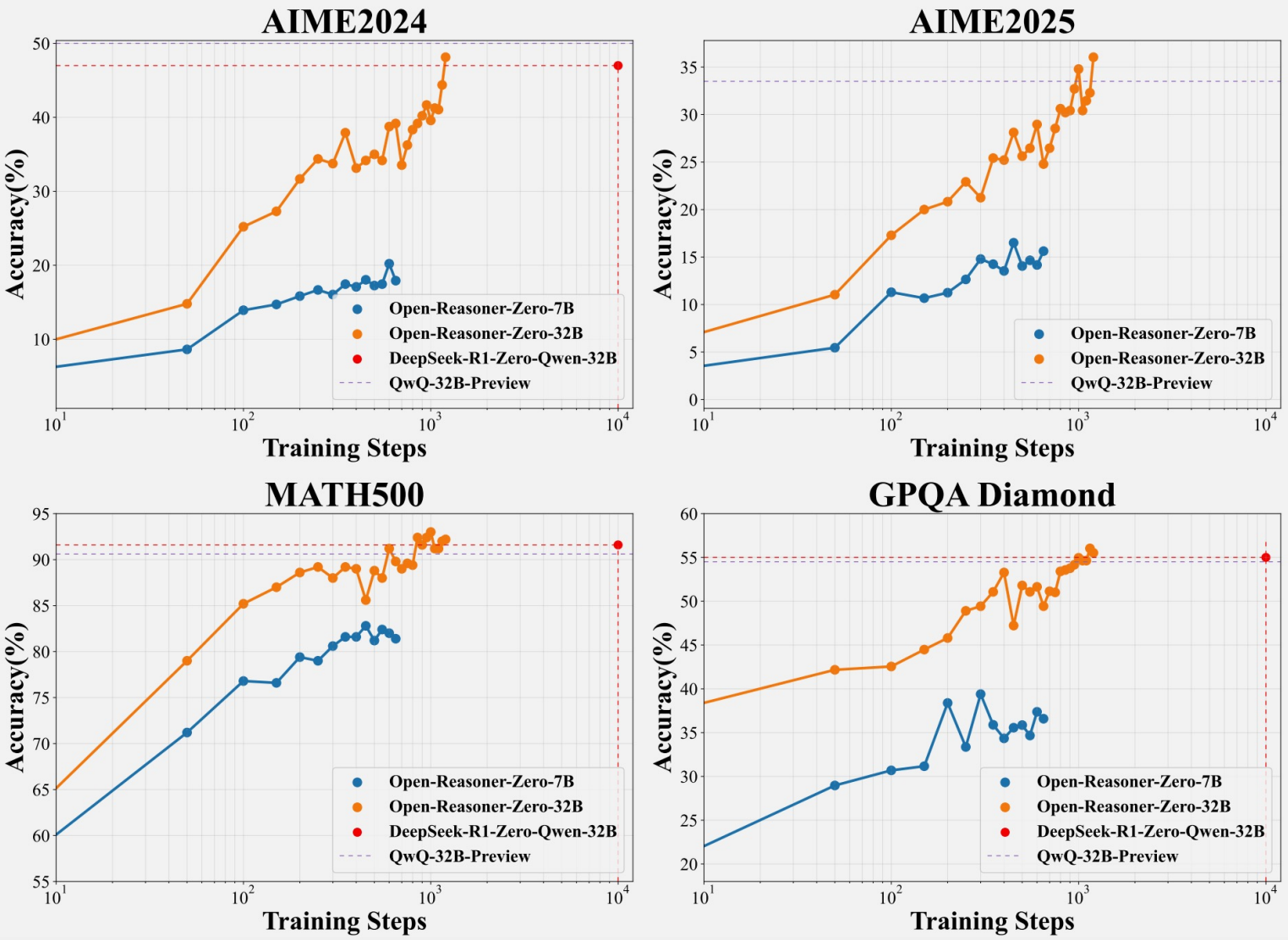
- Why is the KL divergence typically used to force the policy model to remain close to the reference model in RLHF but it does not help here?

J. Hu et al., Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model, 2025

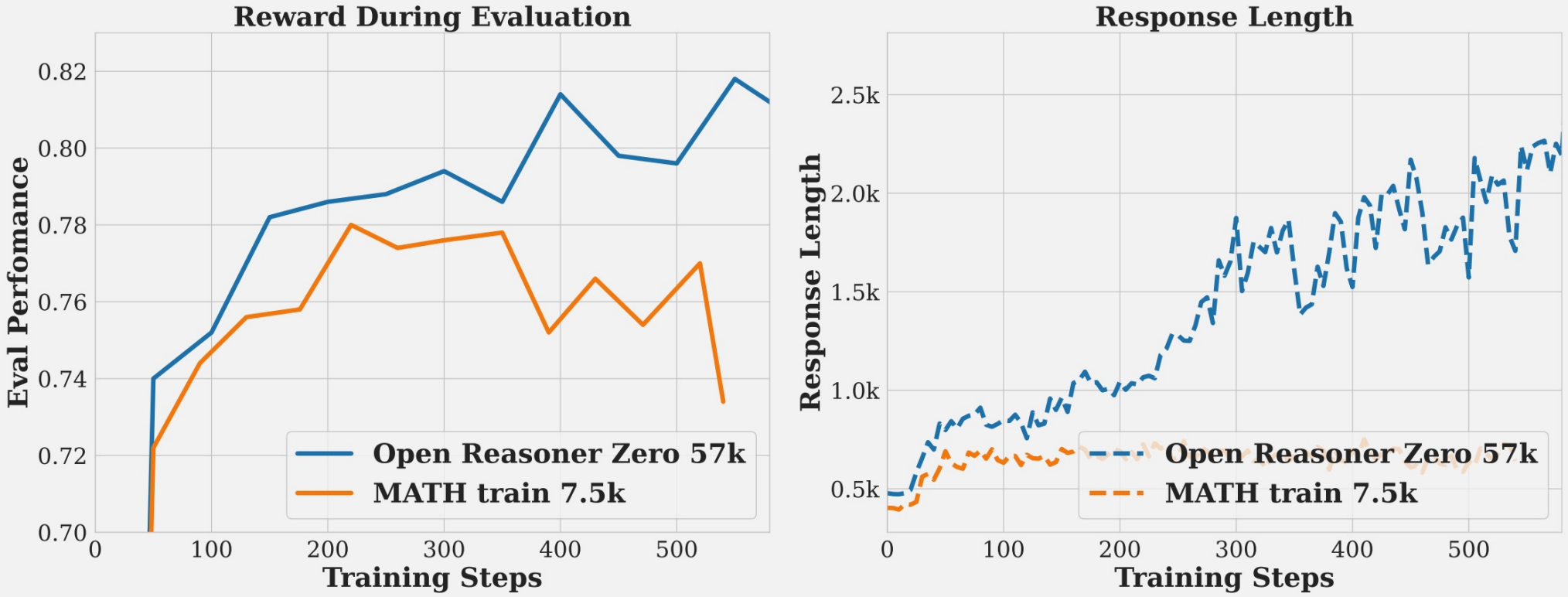
Collecting a Curated Dataset

- ~129k samples spanning mathematics and reasoning domains.
  - includes various sources of problems
  - synthesizes additional reasoning tasks using programmatic approaches to augment the dataset
  - excludes problems that are challenging for rule-based reward function, such as multiple-choice and proof-oriented problems
  - An LLM assesses the pass rate of each problem, removing samples with either too high or zero pass rates.

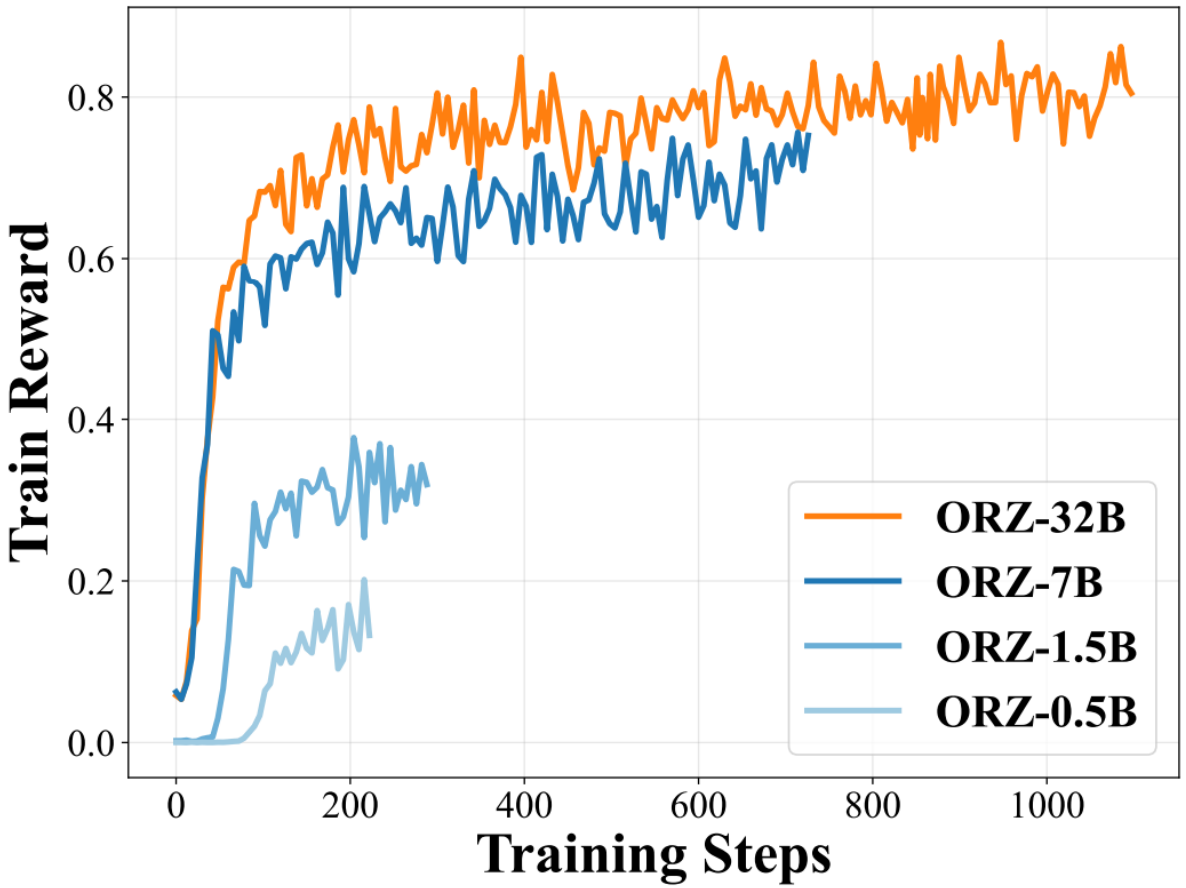
- This collection is carefully balances quantity, diversity, and quality.

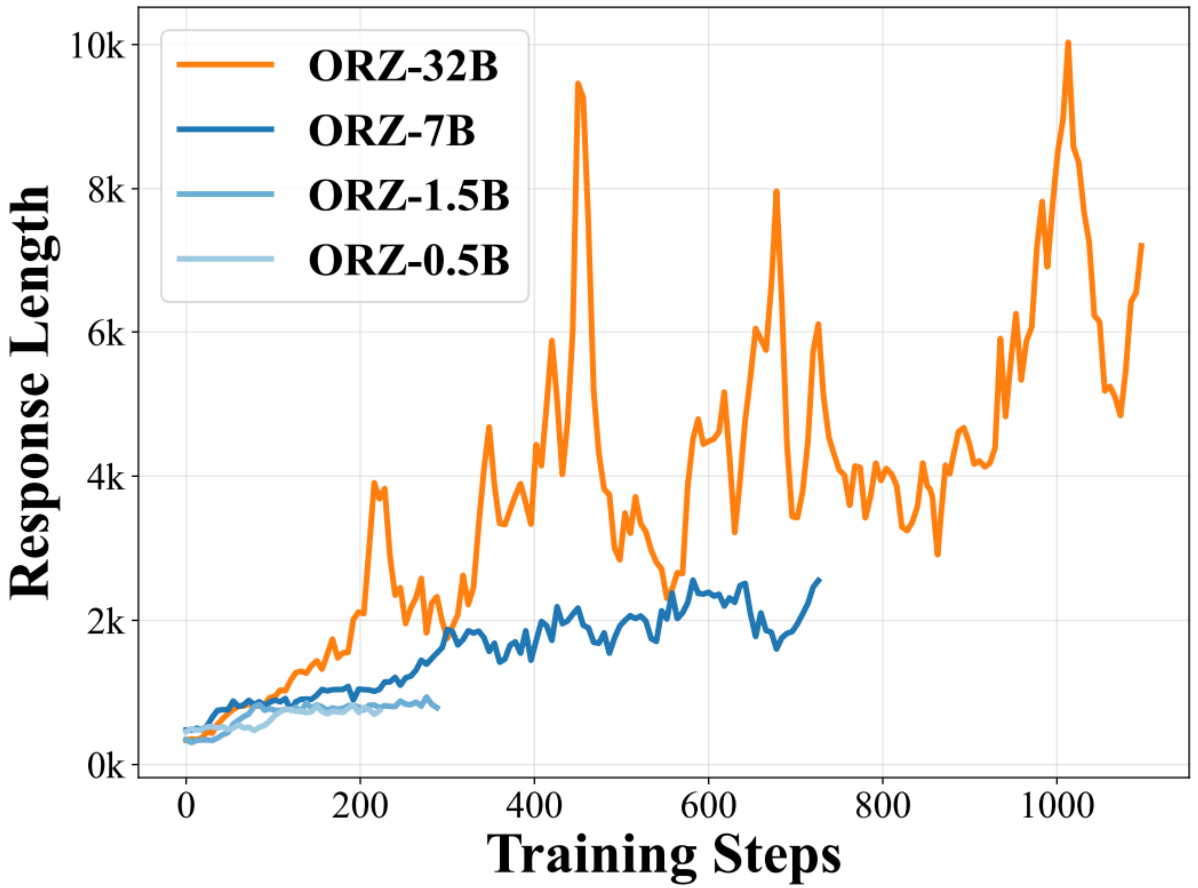


J. Hu et al., Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model, 2025



J. Hu et al., Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model, 2025





J. Hu et al., Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model, 2025

Summary

- Learning reasoning models by RL
  - DeepSeek-R1
    - GRPO
  - DAPO
  - ORZ