北京大學
PEKING UNIVERSITY

GRPO: Group Relative Policy Optimization

Apr 30, 2025

PART 01

Review: PPO and DPO

Proximal Policy Optimization (PPO)

北京大学 PEKING UNIVERSITY

- Goal: Optimize a language model to **adhere to human preferences**.
- Process：

1. Obtain a reference model π_{ref} by **supervised fine-tuning** (SFT)
2. Obtain a reward model $r_\varphi(x, y)$.
3. Training a value function with the reward: $r_t = r_\varphi(x, y_{<t}) - \beta \log \frac{\pi_\theta(y_t|x, y_{<t})}{\pi_{\text{ref}}(y_t|x, y_{<t})}$

   And then obtain the advantage function A_t by Generalized Advantage Estimation.
4. Then one can optimized the model by **gradient methods**.

$$\mathcal{L}_{\text{PPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_{\theta_{\text{old}}}} \frac{1}{y} \sum_{t=1}^{|y|} \frac{\pi_\theta(y_t|x, y_{<t})}{\pi_{\theta_{\text{old}}}(y_t|x, y_{<t})} A_t$$

5. Get θ as the old parameter and then repeat step 4.

Direct Preference Optimization (DPO)

北京大学 PEKING UNIVERSITY

- Goal: Optimize a language model to **adhere to human preferences**.
- Process：

1. Obtain a reference model π_{ref} by **supervised fine-tuning** (SFT)
2. Reparameterizes the **reward function**:

$$r(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x)$$

3. Taking $r(x, y)$ into Bradley Terry (BT) ranking objective:

$$p(y_w \succ y_l \mid x) = \sigma\left(r(x, y_w) - r(x, y_l)\right)$$

one can obtain the **loss function** of DPO:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

4. Then one can optimized the model by **gradient methods**.

PART 02

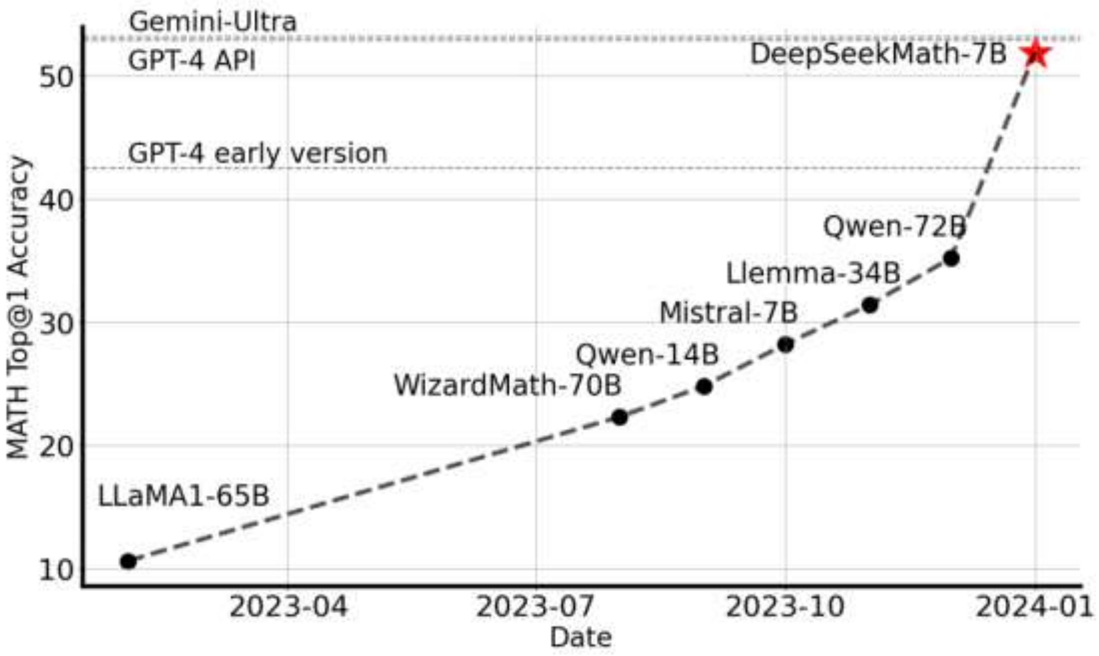
GRPO: Group Relative Policy Optimization

Background: LLM for math reasoning

DeepSeekMath-7B： outperform other math reasoning models on competition-level MATH benchmark.

- Techniques:

  1. **Data selection pipeline** harness the potential of publicly available web data.
  2. **GRPO** technique for reinforce learning.

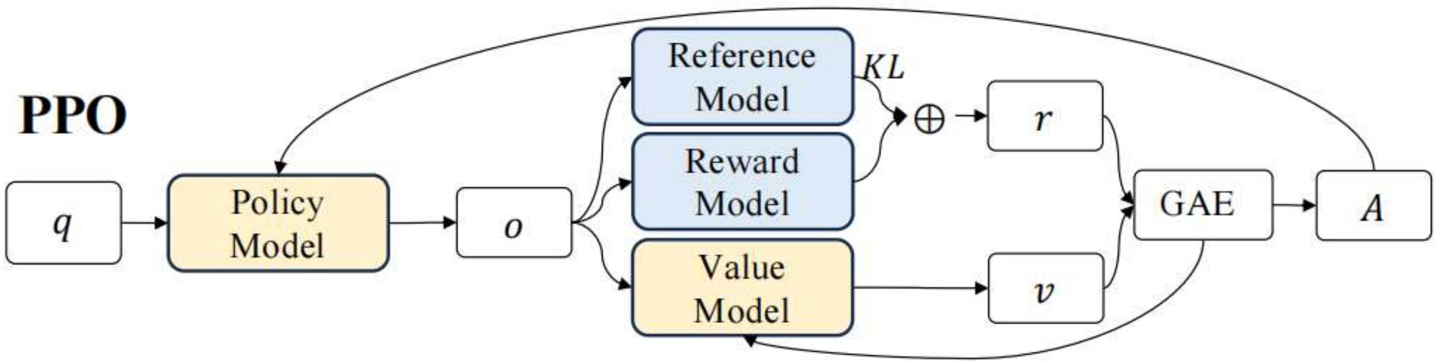


Limitations of PPO

L1: Memory and computational burden. Value function is typically a model of comparable size as the policy model.

L2: Complicate in value function training. In the LLM context, usually only the last token is assigned a reward score by the reward model.

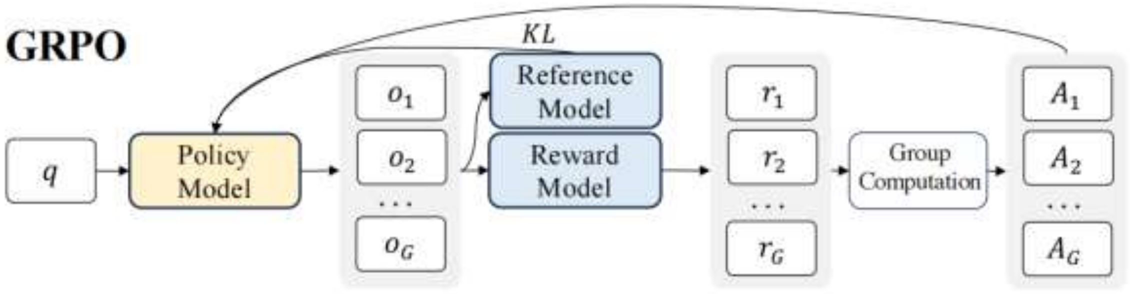
→ ➢ A straightforward motivation: Develop a **value-model-free** framework?



GRPO objective function: Overview

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\boxed{\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(x)}}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\left(\frac{\pi_\theta(y_{i,t}|x_i,y_{i,<t})}{\pi_{\text{old}}(y_{i,t}|x_i,y_{i,<t})}\boxed{\hat{A}_{i,t}} - \beta\boxed{\mathbb{D}_{KL}(\pi_\theta||\pi_{\text{ref}})}\right)\right]$$

- **Value-model-free.**
- **Group outputs**: Sampled from old policy.
- **Group advantage**: Calculated on **relative** rewards inside each group only. Avoid computation of KL-Divergence between reference model and reward model.
- **Approximated KL**: Can viewed as a normalization term. Use a positive and unbiased estimator with little variance: $\mathbb{D}_{KL}(\pi_\theta||\pi_{\text{ref}}) = \frac{\pi_\theta(y_{i,t}|x_i,y_{i,<t})}{\pi_{\text{ref}}(y_{i,t}|x_i,y_{i,<t})} - \log\frac{\pi_\theta(y_{i,t}|x_i,y_{i,<t})}{\pi_{\text{ref}}(y_{i,t}|x_i,y_{i,<t})} - 1$



Group advantage with outcome supervision

- A direct question: How to obtain the advantage? → different for inoutcome/process supervision

- In outcome supervision: **One output, one reward.**
  - Suppose the reward of $\{r_1, r_2, \cdots, r_G\}$ is denoted as $\mathbf{r} = \{r_1, r_2, \cdots, r_G\}$
  - The relative advantage

$$\hat{A}_{i,t} = \tilde{r}_i := \frac{r_i - mean(\mathbf{r})}{std(\mathbf{r})}$$

- High reward in the group → high advantage → more significance in objective function.

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x \sim \mathcal{D}, \{y_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}(x)}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \left(\frac{\pi_\theta(y_{i,t}|x_i, y_{i,<t})}{\pi_{\text{old}}(y_{i,t}|x_i, y_{i,<t})} \boxed{\hat{A}_{i,t}} - \beta \mathbb{D}_{KL}(\pi_\theta || \pi_{\text{ref}}) \right) \right]$$

Group advantage with process supervision

- In outcome supervision: **One output, a series of reward.**
  - Suppose the reward of $\{r_1, r_2, \cdots, r_G\}$ is denoted as $\mathbf{r} = \{\{r_1^1, r_1^2, \cdots, r_1^{index(K_1)}\}, \{r_2^1, r_2^2, \cdots, r_2^{index(K_2)}\}, \cdots, \{r_G^1, r_G^2, \cdots, r_G^{index(K_G)}\}\}$
  - The relative advantage

$$\hat{A}_{i,t} = \boxed{\sum_{index(j) \geq t}} \tilde{r}_i^{index(j)}, \quad \tilde{r}_i^{index(j)} := \frac{r_i^{index(j)} - mean(\mathbf{r})}{std(\mathbf{r})}$$

➢ High total reward for subsequent tokens → high advantage → more significance in objective function.

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x \sim \mathcal{D}, \{y_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}(x)}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \left(\frac{\pi_\theta(y_{i,t}|x_i, y_{i,<t})}{\pi_{\text{old}}(y_{i,t}|x_i, y_{i,<t})} \boxed{\hat{A}_{i,t}} - \beta \mathbb{D}_{KL}(\pi_\theta || \pi_{\text{ref}}) \right) \right]$$

Group advantage with iteration

Algorithm 1 Iterative Group Relative Policy Optimization

Input initial policy model $\pi_{\theta_{\text{init}}}$; reward models r_φ; task prompts $\mathcal{D}$; hyperparameters ε, β, μ

```
policy model π_θ ← π_θinit
for iteration = 1, . . . , I do
    reference model π_ref ← π_θ
    for step = 1, . . . , M do
        Sample a batch D_b from D
        Update the old policy model π_θold ← π_θ
        Sample G outputs {o_i}_{i=1}^G ~ π_θold(· | q) for each question q ∈ D_b
        Compute rewards {r_i}_{i=1}^G for each sampled output o_i by running r_φ
        Compute Â_{i,t} for the t-th token of o_i through group relative advantage estimation.
        for GRPO iteration = 1, . . . , μ do
            Update the policy model π_θ by maximizing the GRPO objective (Equation 21)
    Update r_φ through continuous training using a replay mechanism.
```

Output π_θ

$$GC_{GRPO}(q, o, t, \pi_{\theta_{rm}}) = \hat{A}_{i,t} + \beta \left(\frac{\pi_{ref}(o_{i,t}|o_{i,<t})}{\pi_\theta(o_{i,t}|o_{i,<t})} - 1 \right), \tag{21}$$

- Update reference model by current policy model. Skip the SFT process.

Experimental results

北京大学 PEKING UNIVERSITY

Model	Size	English Benchmarks		Chinese Benchmarks	
		GSM8K	MATH	MGSM-zh	CMATH
Chain-of-Thought Reasoning					
Closed-Source Model					
Gemini Ultra	-	94.4%	53.2%	-	-
GPT-4	-	92.0%	52.9%	-	86.0%
Inflection-2	-	81.4%	34.8%	-	-
GPT-3.5	-	80.8%	34.1%	-	73.8%
Gemini Pro	-	86.5%	32.6%	-	-
Grok-1	-	62.9%	23.9%	-	-
Baichuan-3	-	88.2%	49.2%	-	-
GLM-4	-	87.6%	47.9%	-	-
Open-Source Model					
InternLM2-Math	20B	82.6%	37.7%	-	-
Qwen	72B	78.9%	35.2%	-	-
Math-Shepherd-Mistral	7B	84.1%	33.0%	-	-
WizardMath-v1.1	7B	83.2%	33.0%	-	-
DeepSeek-LLM-Chat	67B	84.1%	32.6%	74.0%	80.3%
MetaMath	70B	82.3%	26.6%	66.4%	70.9%
SeaLLM-v2	7B	78.2%	27.5%	64.8%	-
ChatGLM3	6B	72.3%	25.7%	-	-
WizardMath-v1.0	70B	81.6%	22.7%	64.8%	65.4%
DeepSeekMath-Instruct	7B	82.9%	46.8%	73.2%	84.6%
DeepSeekMath-RL	7B	**88.2%**	**51.7%**	**79.6%**	**88.8%**

Model	Size	English Benchmarks		Chinese Benchmarks	
		GSM8K	MATH	MGSM-zh	CMATH
Tool-Integrated Reasoning					
Closed-Source Model					
GPT-4 Code Interpreter	-	97.0%	69.7%	-	-
Open-Source Model					
InternLM2-Math	20B	80.7%	54.3%	-	-
DeepSeek-LLM-Chat	67B	86.7%	51.1%	76.4%	85.4%
ToRA	34B	80.7%	50.8%	41.2%	53.4%
MAmmoTH	70B	76.9%	41.8%	-	-
DeepSeekMath-Instruct	7B	83.7%	57.4%	72.0%	84.3%
DeepSeekMath-RL	7B	**86.7%**	**58.8%**	**78.4%**	**87.6%**

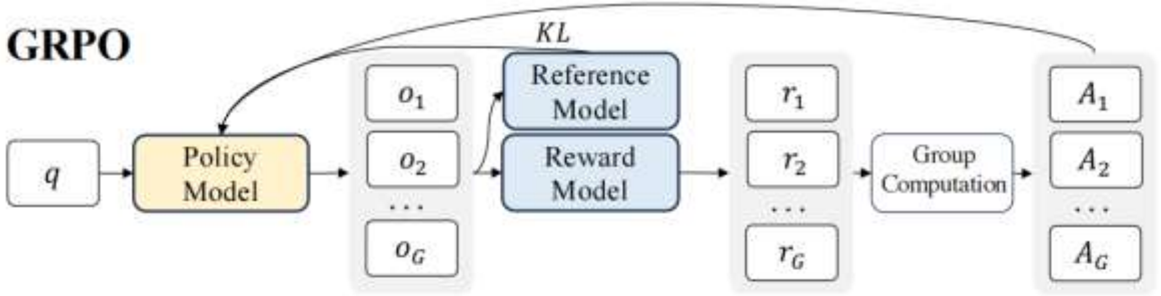
GRPO: A quick summary

- Core technique: Group advantage.
- **Equals to approximate the value function according to the sampled group of model**

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}, \boxed{\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}(x)}}} \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\left(\frac{\pi_\theta(y_{i,t}|x_i, y_{i,<t})}{\pi_{\text{old}}(y_{i,t}|x_i, y_{i,<t})}\boxed{\hat{A}_{i,t}} - \beta\boxed{\mathbb{D}_{KL}(\pi_\theta||\pi_{\text{ref}})} \right)\right]$$

- **Value-model-free.**
- **Group outputs**: Sampled from old policy.
- **Group advantage**: Calculated on **relative** rewards inside each group only. Avoid computation of KL-Divergence.
- **Approximated KL**: Use a positive and unbiased estimator with little variance:

$$\mathbb{D}_{KL}(\pi_\theta||\pi_{\text{ref}}) = \frac{\pi_\theta(y_{i,t}|x_i, y_{i,<t})}{\pi_{\text{ref}}(y_{i,t}|x_i, y_{i,<t})} - \log\frac{\pi_\theta(y_{i,t}|x_i, y_{i,<t})}{\pi_{\text{ref}}(y_{i,t}|x_i, y_{i,<t})} - 1$$



北京大學 PEKING UNIVERSITY

PART 03

Insights from reinforcement learning.

A unified paradigm for different training methods

北京大学 PEKING UNIVERSITY

- For typical reinforcement training algorithms (SFT, RFT, PPO, DPO, GRPO), the gradient of the parameter has the same formulation:

$$\nabla_\theta \mathcal{J}_{\mathcal{A}}(\theta) = \mathbb{E}[\underbrace{(q,o) \sim \mathcal{D}}_{Data\ Source}] \left(\frac{1}{|o|} \sum_{t=1}^{|o|} \underbrace{GC_{\mathcal{A}}(q,o,t,\pi_{rf})}_{Gradient\ Coefficient} \nabla_\theta \log \pi_\theta(o_t|q,o_{<t}) \right).$$

- Three core components:

1. Data source $\mathcal{D}$.
2. Reward function π_{rf} : a **rule** not changing during training / a **model** changing during training
3. Algorithm $\mathcal{A}$: processes the **training data** and the **reward signal** to the **gradient coefficient**, determines the **magnitude** of the penalty or reinforcement.

A unified paradigm for different training methods

➢ Three core components for existing algorithms:

Methods	Data Source	Reward Function	Gradient Coefficient
SFT	$q, o \sim P_{sft}(Q, O)$	-	1
RFT	$q \sim P_{sft}(Q), o \sim \pi_{sft}(O\|q)$	Rule	Equation $\boxed{10}$
DPO	$q \sim P_{sft}(Q), o^+, o^- \sim \pi_{sft}(O\|q)$	Rule	Equation $\boxed{14}$
Online RFT	$q \sim P_{sft}(Q), o \sim \pi_{\theta}(O\|q)$	Rule	Equation $\boxed{10}$
PPO	$q \sim P_{sft}(Q), o \sim \pi_{\theta}(O\|q)$	Model	Equation $\boxed{18}$
GRPO	$q \sim P_{sft}(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta}(O\|q)$	Model	Equation $\boxed{21}$

- The detailed Equations can be found in the original paper.

How data source affect the training performance

PEKING UNIVERSITY

- ➢ Online sampling: Sampling distribution can change during training *(e.g. PPO, GRPO)*.
- ➢ Offline sampling: Sampling distribution is fixed during training *(e.g. RFT, DPO)*.

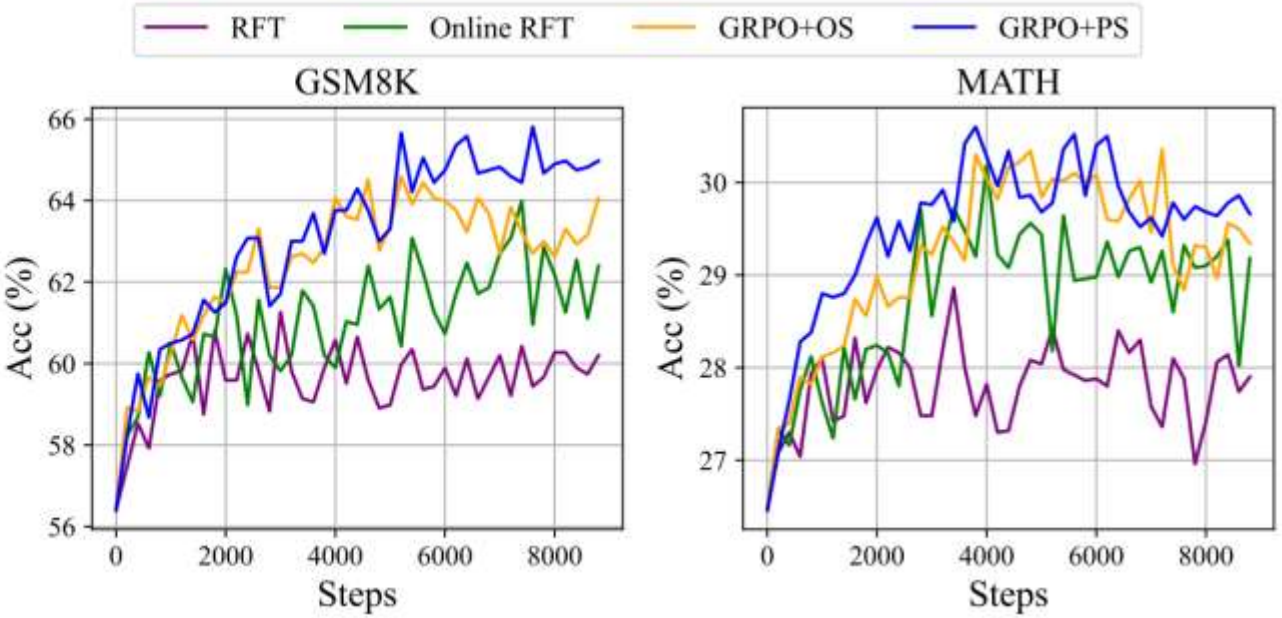


Figure 5 | Performance of the DeepSeekMath-Instruct 1.3B model, which was further trained using various methods, on two benchmarks.

- Online sampling performs well in the later stage.

Impact on reward function and gradient coefficients

➢ Model vs Rule:

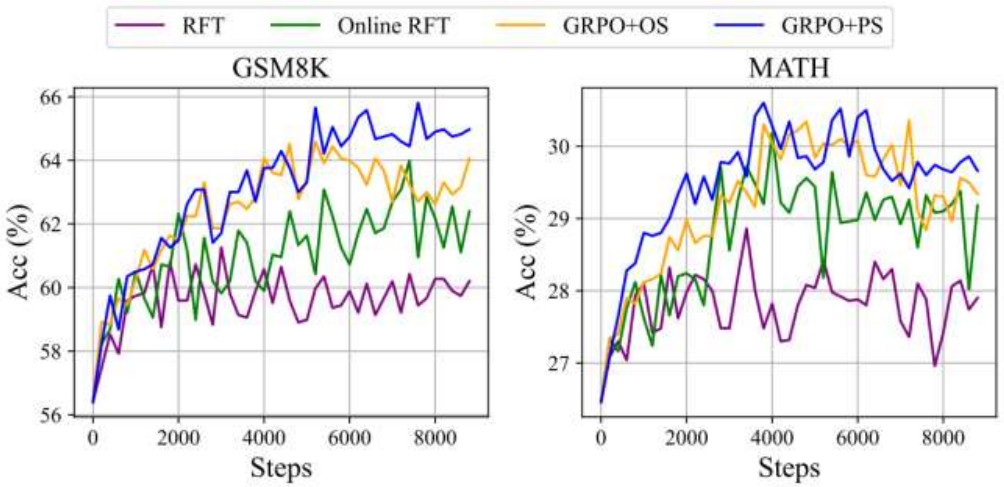


Figure 5 | Performance of the DeepSeekMath-Instruct 1.3B model, which was further trained using various methods, on two benchmarks.

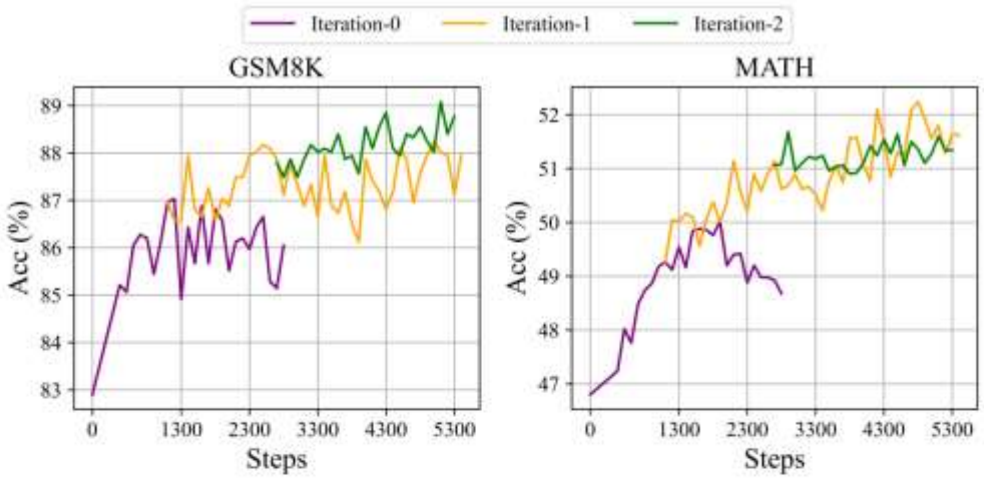
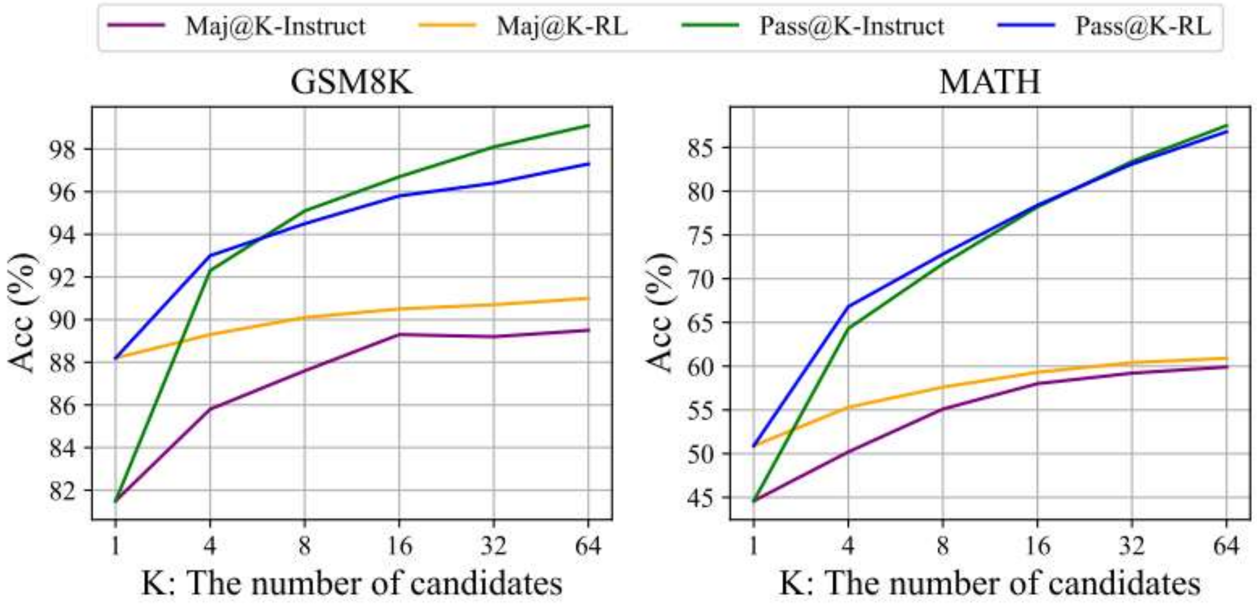


Figure 6 | Performance of iterative reinforcement learning with DeepSeekMath-Instruct 7B on two benchmarks.

- GRPO vs RFT: Model is better than rule.
- GPRO+OS vs GPRO+PS: fine-grained, step-aware gradient coefficients benefits.
- Iteration can improve the training performance, especially in Iteration 1.

Why RL works?

- Instruct tuning model vs RL



- Maj@K: generalize first K answer and take majority vote, the voted one is right.
- Pass@K: generalize K answer, at least one is right.

- RL improves Maj@K but not Pass@K!
- Maybe RL improves the ability for **boosting the correct response from TopK** rather than improves fundamental capabilities.

How to achieve a more efficient RL

PEKING UNIVERSITY 北京大学

- Better performance: using sampling strategy and reward function vary **dynamically**.
- 堆trick?

➢ Potential future directions from authors

1. Data source: Queries are all from the instruction tuning stage with simple sampling strategy. → exploration on **out-of-distribution** samples with advance sampling strategy.
2. Algorithms: Fully trust the reward function, impossible to guarantee the reward signal is always reliable → develop algorithms **robust against noisy reward** signals.
3. Reward function: (1) enhance the **generalization** ability; (2) reflect the **uncertainty**; (3) build high quality **process reward models**.