GRPO – Group Relative Policy Optimization

Yu Chung (Paul) Lee, Wentai Hu

4 March 2026

DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song,
Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y.K. Li, Y. Wu, Daya Guo
DeepSeek-AI, Tsinghua University, Peking University

{zhihongshao,wangpeiyi,zhuqh,guoday}@deepseek.com
https://github.com/deepseek-ai/DeepSeek-Math

Background

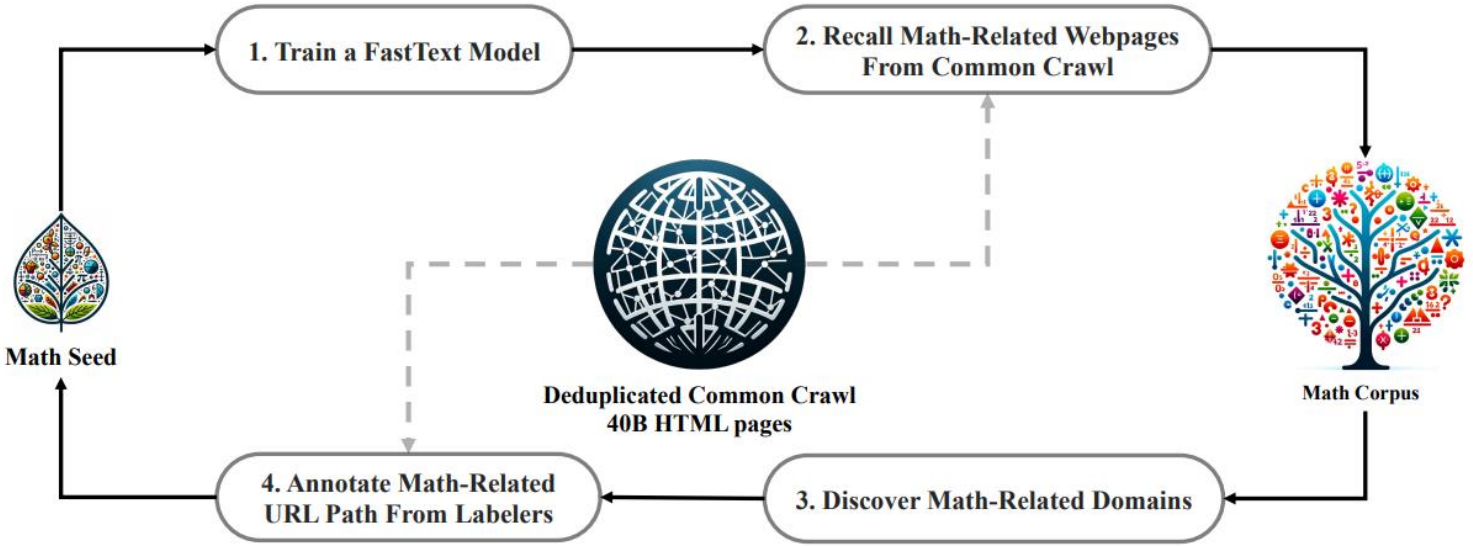
LLMs have achived excellent performance in general language tasks. They also changed the way AI approaches mathematical reasoning. However, the strongest mathematical reasoning models today are mostly closed-source, such as GPT-4 and Gemini, leaving a significant gap for the open source community. This motivates the development of DeepSeekMath, an open model designed for strong mathematical reasoning.

Key Contributions

- DeepSeekMath Corpus
- Three Model Versions:
  - DeepSeekMath-Base 7B (Math Pre-training)
  - DeepSeekMath-Instruct 7B (SFT)
  - DeepSeekMath-RL 7B (RL)
- Group Relative Policy Optimization Algorithm (GRPO)
- Performance Evaluation

Math Pre-training

Data Collection pipeline and Decontamination



Math Pre-training

Validating the Quality of the DeepSeekMath Corpus

Training Setup:

- Base model: DeepSeek-LLM 1.3B
- Training 150B tokens on each math corpus
- Compare performance across multiple math benchmarks

Optimizer	AdamW
Beta1,2	0.9,0.95
Max LR	5.3e-4
Batch Size	4M tokens
Context	4K

Math Pre-training

Validating the Quality of the DeepSeekMath Corpus

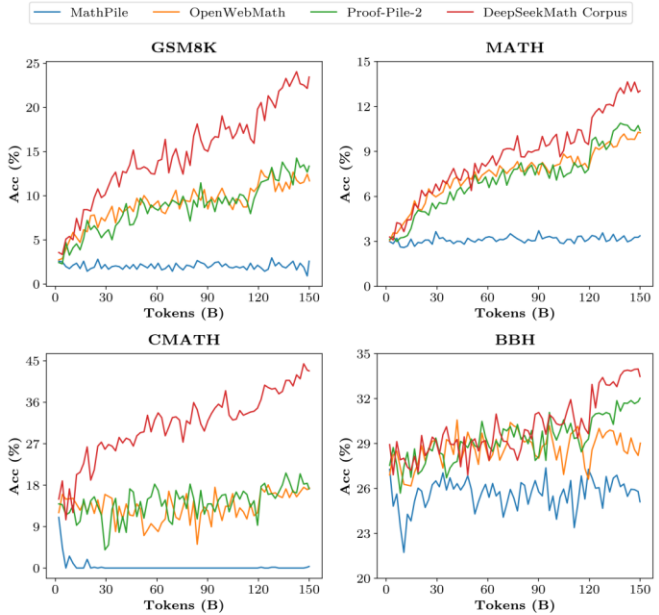


Figure 3 | Benchmark curves of DeepSeek-LLM 1.3B trained on different mathematical corpora.

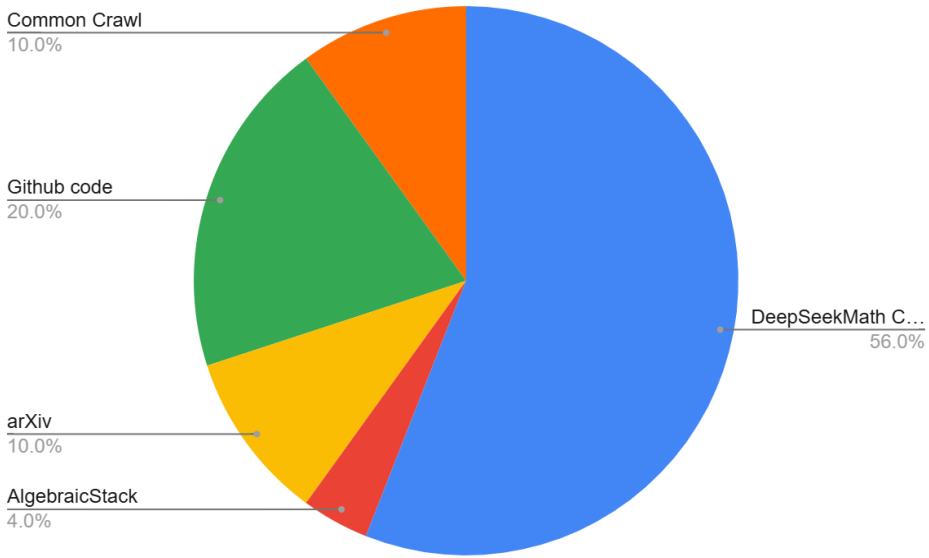
Math Corpus	Size	English Benchmarks					Chinese Benchmarks		
		GSM8K	MATH	OCW	SAT	MMLU STEM	CMATH	Gaokao MathCloze	Gaokao MathQA
No Math Training	N/A	2.9%	3.0%	2.9%	15.6%	19.5%	12.3%	0.8%	17.9%
MathPile	8.9B	2.7%	3.3%	2.2%	12.5%	15.7%	1.2%	0.0%	2.8%
OpenWebMath	13.6B	11.5%	8.9%	3.7%	31.3%	29.6%	16.8%	0.0%	14.2%
Proof-Pile-2	51.9B	14.3%	11.2%	3.7%	43.8%	29.2%	19.9%	5.1%	11.7%
DeepSeekMath Corpus	**120.2B**	**23.8%**	**13.6%**	**4.8%**	**56.3%**	**33.1%**	**41.5%**	**5.9%**	**23.6%**

- High performance across multiple math benchmarks
- Contains multilingual Math Content
- Largest math corpus in size

DeepseekMath-Base 7B

Setup:

- Initialized with DeepSeek-Coder-Base-v1.5 7B
- Trained for 500B tokens
- Data distribution:



Training with DeepseekMath-Base 7B

Math Problem Solving with Step-by-Step Reasoning

Model	Size	English Benchmarks					Chinese Benchmarks		
		GSM8K	MATH	OCW	SAT	MMLU STEM	CMATH	Gaokao MathCloze	Gaokao MathQA
Closed-Source Base Model									
Minerva	7B	16.2%	14.1%	7.7%	-	35.6%	-	-	-
Minerva	62B	52.4%	27.6%	12.0%	-	53.9%	-	-	-
Minerva	540B	58.8%	33.6%	17.6%	-	63.9%	-	-	-
Open-Source Base Model									
Mistral	7B	40.3%	14.3%	9.2%	71.9%	51.1%	44.9%	5.1%	23.4%
Llemma	7B	37.4%	18.1%	6.3%	59.4%	43.1%	43.4%	11.9%	23.6%
Llemma	34B	54.0%	25.3%	10.3%	71.9%	52.9%	56.1%	11.9%	26.2%
DeepSeekMath-Base	7B	**64.2%**	**36.2%**	**15.4%**	**84.4%**	**56.5%**	**71.7%**	**20.3%**	**35.3%**

Table 2 | Comparisons between DeepSeekMath-Base 7B and strong base models on English and Chinese mathematical benchmarks. Models are evaluated with chain-of-thought prompting. Minerva results are quoted from Lewkowycz et al. (2022a).

Training with DeepseekMath-Base 7B

Math Problem Solving with Tool Use and Formal Proving

Model	Size	Problem Solving w/ Tools		Informal-to-Formal Proving	
		GSM8K+Python	MATH+Python	miniF2F-valid	miniF2F-test
Mistral	7B	48.5%	18.2%	18.9%	18.0%
CodeLlama	7B	27.1%	17.2%	16.3%	17.6%
CodeLlama	34B	52.7%	23.5%	18.5%	18.0%
Llemma	7B	41.0%	18.6%	20.6%	22.1%
Llemma	34B	64.6%	26.3%	21.0%	21.3%
DeepSeekMath-Base	7B	**66.9%**	**31.4%**	**25.8%**	**24.6%**

Natural Language Understanding, Reasoning, and Code

Model	Size	MMLU	BBH	HumanEval (Pass@1)	MBPP (Pass@1)
Mistral	7B	**62.4%**	55.7%	28.0%	41.4%
DeepSeek-Coder-Base-v1.5†	7B	42.9%	42.9%	40.2%	52.6%
DeepSeek-Coder-Base-v1.5	7B	49.1%	55.2%	**43.2%**	**60.4%**
DeepSeekMath-Base	7B	54.9%	**59.5%**	40.9%	52.6%

DeepSeekMath-Instruct 7B

Dataset:

- 776K math instruction examples
- English and Chinese meth problems (GSM8K, MATH)
- Covers topics such as algebra, probability, number theory, calculus, and geometry

Reasoning Formats:

- Chain-of-Thought (CoT)
- Program-of-Thought (PoT)
- Tool-integrated reasoning

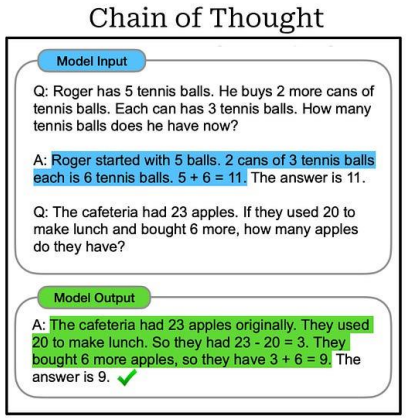
Training:

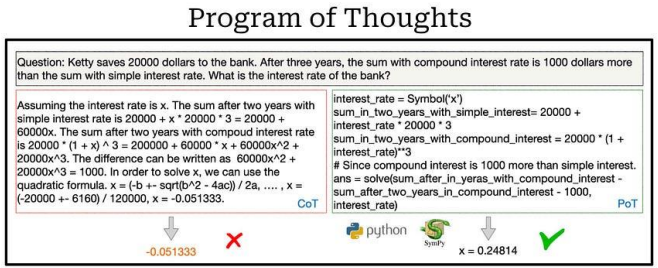
- Fine-tuned from DeepSeekMath-Base 7B
- Training sequences concatenated up to 4K context length

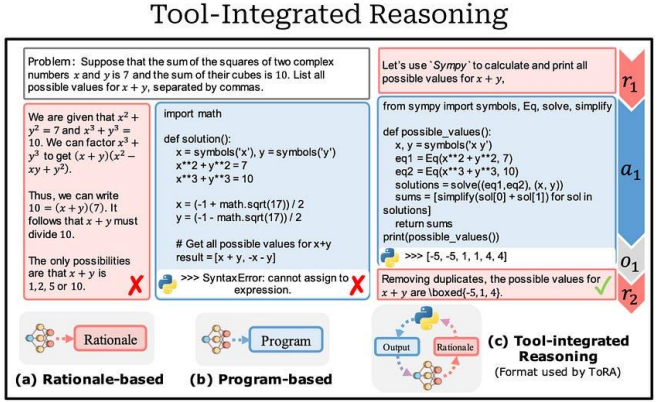
Model	Size	English Benchmarks		Chinese Benchmarks	
		GSM8K	MATH	MGSM-zh	CMATH
Chain-of-Thought Reasoning					
Closed-Source Model					
Gemini Ultra	-	94.4%	53.2%	-	-
GPT-4	-	92.0%	52.9%	-	86.0%
Inflection-2	-	81.4%	34.8%	-	-
GPT-3.5	-	80.8%	34.1%	-	73.8%
Gemini Pro	-	86.5%	32.6%	-	-
Grok-1	-	62.9%	23.9%	-	-
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%
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%

Solution SFT

Solution SFT dataset are created using three different formats



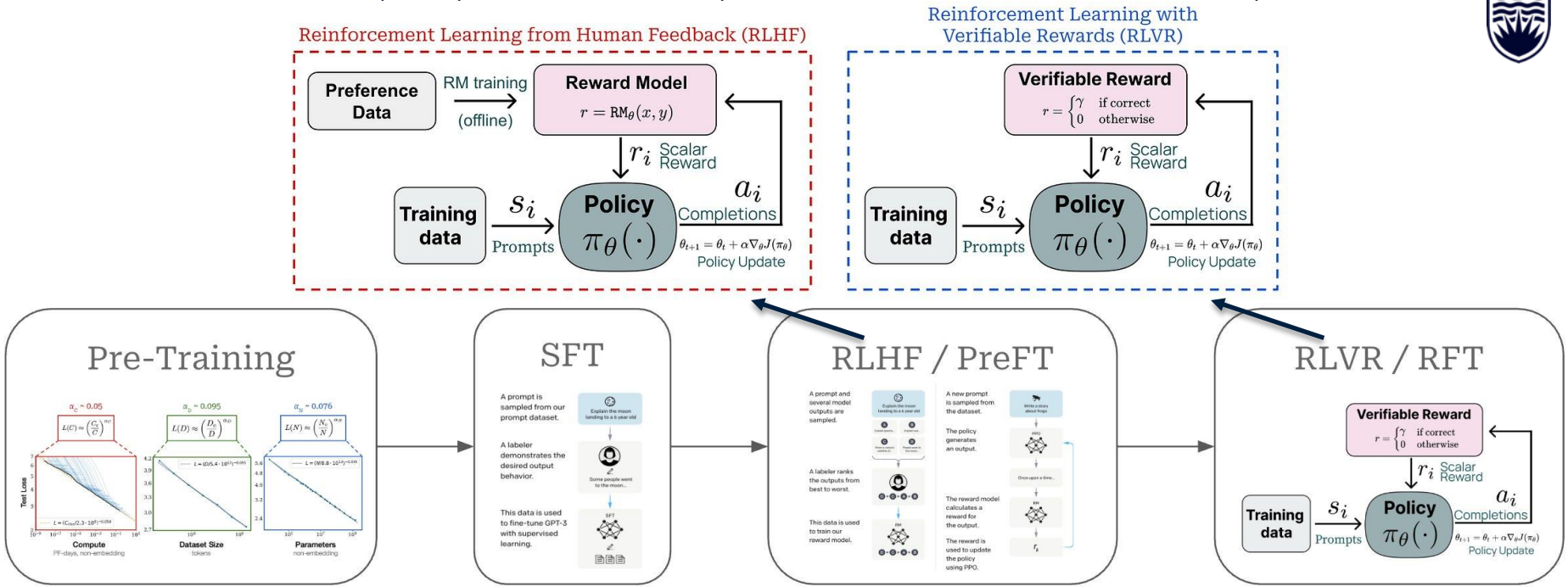




Reinforcement Learning for LLMs

RL from Human Feedback (RLHF) – Neural Reward (derived from a human preference reward model)

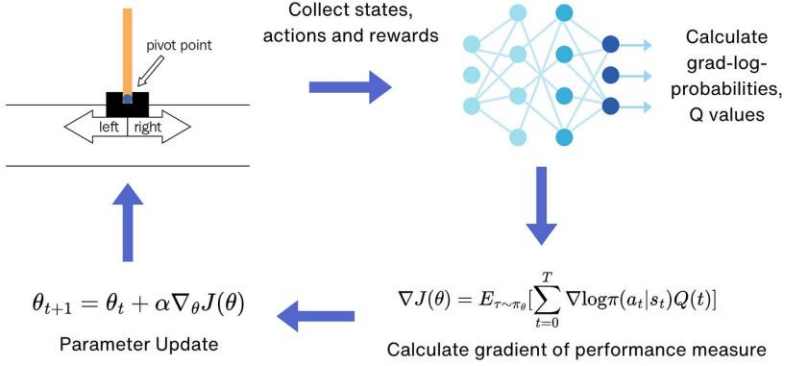
RL with Verifiable Rewards (RLVR) – Verifiable Reward (derived from rules-based or heuristic verifier)

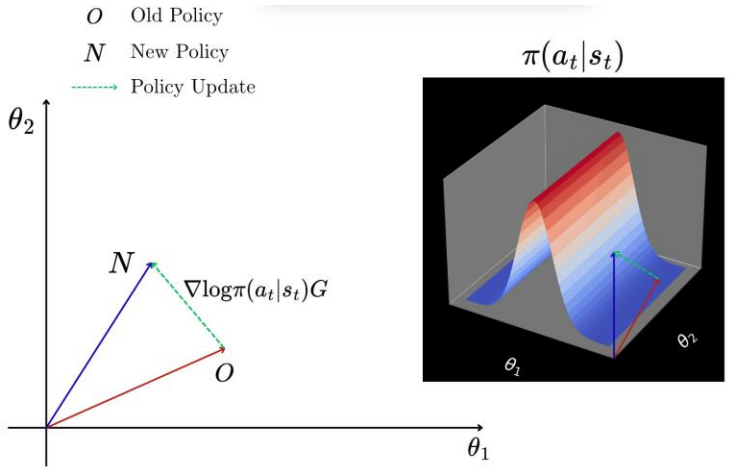


Deep (Learning) Focus - Group Relative Policy Optimization (GRPO). Cameron R. Wolfe (2025). Group Relative Policy Optimization (GRPO)

Policy Gradient Methods in Reinforcement Learning

Policy Gradient Methods for Reinforcement Learning with Function Approximation (Sutton, 1999)





$$\nabla_\theta J(\pi_\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, A^{\pi_\theta}(s_t, a_t)\right]$$

Advantage function (used in the surrogate loss)

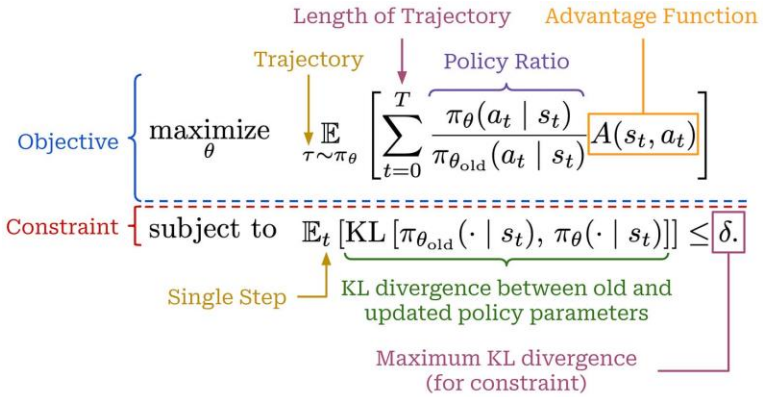
Value function (predicted by the critic)

$$\overbrace{A(s_t, a_t)} = \underbrace{Q(s_t, a_t)} - \overbrace{V(s_t)}$$

State-action value function (equal to observed reward)

Policy Gradient Methods in Reinforcement Learning. Vizuara AI (2025). Policy Gradient Methods in Reinforcement Learning

From TRPO to PPO



Trust Region Policy Optimization

Proximal Policy Optimization

- Same surrogate objective but using clipping instead of hard KL constraint
- Simple, stable, efficient, first-order

Policy Ratio $r_t(\theta)$

$$\mathcal{J}_{PPO}(\theta) = \mathbb{E}[q \sim P(Q), o \sim \pi_{\theta_{old}}(O|q)] \frac{1}{|o|}\sum_{t=1}^{|o|}\min\left[\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})}A_t, \text{clip}\left(\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})}, 1-\varepsilon, 1+\varepsilon\right)A_t\right]$$

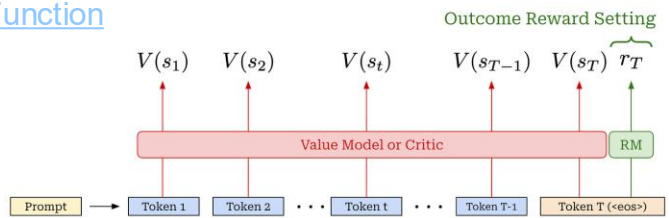
$$L_t^{\text{CLIP}}(\theta) = \begin{cases} r_t(\theta)\,A_t & \text{Case \#1: } A_t > 0,\ r_t(\theta) \leq 1+\epsilon \\ (1+\epsilon)\,A_t & \text{Case \#2: } A_t > 0,\ r_t(\theta) > 1+\epsilon \\ r_t(\theta)\,A_t & \text{Case \#3: } A_t < 0,\ r_t(\theta) \geq 1-\epsilon \\ (1-\epsilon)\,A_t & \text{Case \#4: } A_t < 0,\ r_t(\theta) < 1-\epsilon \end{cases}$$

Reward Model

$$r'(x,y) = \overbrace{r(x,y)}^{\text{Original Reward}} - \overbrace{\beta\mathbb{D}_{\text{KL}}(\pi(y|x) \parallel \pi_{\texttt{ref}}(y|x))}^{\text{KL Divergence}}$$

Scaling Factor

Value Function (Critic)



Deep (Learning) Focus - PPO for LLMs: A Guide for Normal People. Cameron R. Wolfe (2025). PPO for LLMs: A Guide for Normal People

Policy Ratio (Important Ratio)

Clipped Surrogate Objective

Let $r_t(\theta)$ denote the probability ratio $r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$, so $r(\theta_{\text{old}}) = 1$. TRPO maximizes a "surrogate" objective

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right] = \hat{\mathbb{E}}_t\left[r_t(\theta)\hat{A}_t\right]. \tag{6}$$

$$\begin{aligned}
\nabla L^{PG} &= \mathbb{E}_{a_t,s_t\sim\pi_\theta}\left[\nabla \log \pi_\theta(a_t|s_t)A_t\right] \\
&= \int_{a,s} \pi_\theta(a|s)\nabla \log \pi_\theta(a|s)A_t \\
&= \int_{a,s} \frac{\pi_{\theta_{old}}(a|s)}{\pi_{\theta_{old}}(a|s)}\pi_\theta(a|s)\frac{\nabla\pi_\theta(a|s)}{\pi_\theta(a|s)}A_t \\
&= \mathbb{E}_{a_t,s_t\sim\pi_{\theta_{old}}}\left[\frac{\nabla\pi_\theta(a_s|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}A_t\right] \\
&= \nabla L^{CPI}
\end{aligned}$$

$$L^{PG} = \int \nabla L^{PG} = \mathbb{E}_{a_t,s_t\sim\pi_\theta}\left[\log \pi_\theta(a_t|s_t)A_t\right]$$

$$L^{CPI} = \int \nabla L^{CPI} = \mathbb{E}_{a_t,s_t\sim\pi_{\theta_{old}}}\left[\frac{\pi_\theta(a_s|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}A_t\right]$$

The difference between the two is that in L^{PG} states and actions are sampled using the original policy π_θ. This means that once you perform one gradient update step you have to throw away the data and collect new.

In L^{CPI} the states and actions are collected under $\pi_{\theta_{old}}$, which means that you can perform multiple updates with the same dataset. Obviously, if you perform only one update step with PPO you will get the vanilla PG. After the first update step the two objectives will differ.

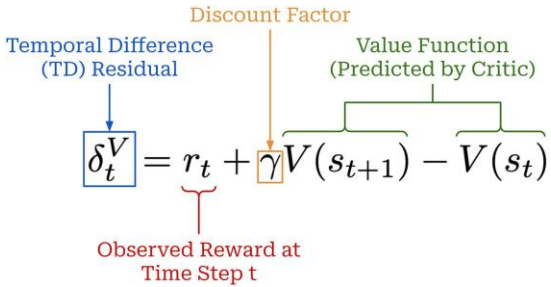
reinforcement learning - Where does the proximal policy optimization objective's ratio term come from? - Artificial Intelligence Stack Exchange

PPO – Generalized Advantage Estimation (GAE)

Advantage function: $A(s_t, a_t) = Q(s_t, a_t) - V(s_t)$ estimated by GAE

Temporal Difference (TD):
one-step estimation using per-token predictions from the critic

GAE:
N-step advantage estimates by a mixing parameter λ
$\lambda = 0$ single-step TD residual
$\lambda = 1$ recovers Monte Carlo estimate



GAE Formula

$$\hat{A}_t^{GAE} = (1-\lambda)\left(\hat{A}_t^{(1)} + \lambda \hat{A}_t^{(2)} + \lambda^2 \hat{A}_t^{(3)} + \cdots\right)$$

Expand Definition

$$= (1-\lambda)\left(\delta_t^V + \lambda(\delta_t^V + \gamma\delta_{t+1}^V) + \lambda^2(\delta_t^V + \gamma\delta_{t+1}^V + \gamma^2\delta_{t+2}^V) + \cdots\right)$$

$$= (1-\lambda)\left(\delta_t^V(1 + \lambda + \lambda^2 + \cdots) + \gamma\delta_{t+1}^V(\lambda + \lambda^2 + \cdots) + \cdots\right)$$

Geometric Series Formula

$$= (1-\lambda)\left(\delta_t^V \frac{1}{1-\lambda} + \gamma\delta_{t+1}^V \frac{\lambda}{1-\lambda} + \cdots\right)$$

$$= \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V$$

Exponentially decayed sum of TD residual terms!

$$\hat{A}_t^{GAE} = \sum_{l=0}^{T-t} (\gamma\lambda)^l \, \delta_{t+l}^V$$

$$= \delta_t^V + \gamma\delta_{t+1}^V + \cdots + \gamma^T \delta_T^V$$

$$= (r_t + \gamma V(s_{t+1}) - V(s_t)) + \gamma(r_{t+1} + \gamma V(s_{t+2}) - V(s_{t+1})) + \cdots + \gamma^{T-t}(r_T - V(s_T))$$

$$= (r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t} r_T) - V(s_t)$$

$$= R(s_t) - V(S_t)$$

Deep (Learning) Focus - PPO for LLMs: A Guide for Normal People. Cameron R. Wolfe (2025). PPO for LLMs: A Guide for Normal People

From PPO to Group Relative Policy Optimization (GRPO)

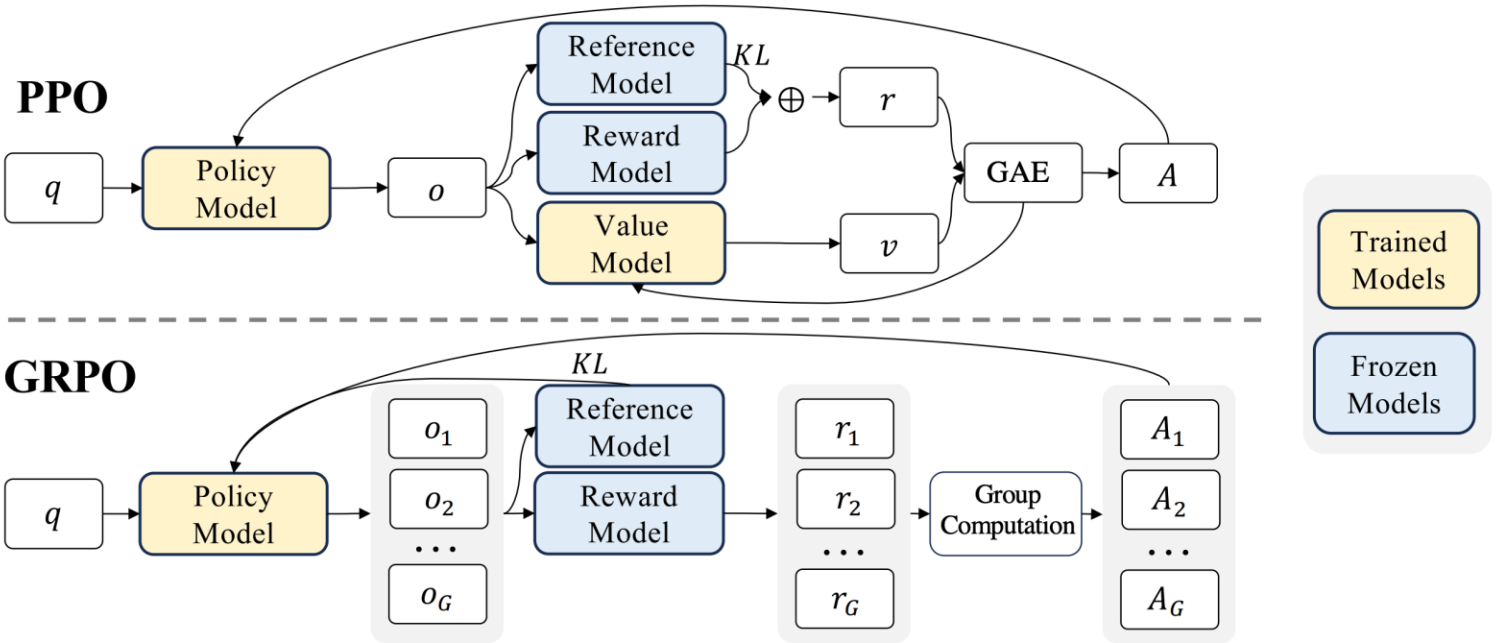
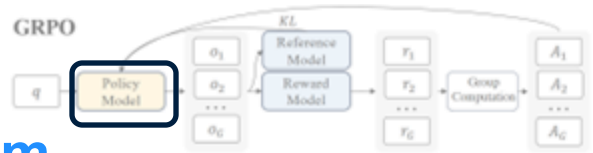


Figure 4 | Demonstration of PPO and our GRPO. GRPO foregoes the value model, instead estimating the baseline from group scores, significantly reducing training resources.

DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. Zhihong Shao, et al (2024). CoRR.



GRPO Objective functions and update algorithm

GRPO Objective function

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

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

Advantage from relative rewards

Length normalization

Policy Ratio: $r_t(\theta)$

KL Estimator

Samples group of outputs $\{o_1, o_2, \ldots, o_G\}$ from the old policy $\pi_{\theta_{old}}$

Policy Ratio:

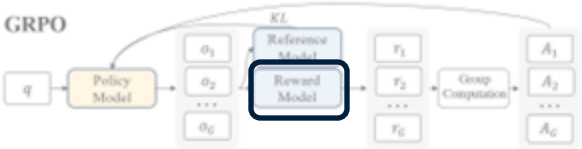
policy ratio = 1 at start, changes on next gradient step for the batch (>1 for +ve advantage or <1 if –ve). Commonly take 1 - 4 gradient steps per batch before updating $\pi_{\theta_{old}}$

Algorithm 1 Iterative Group Relative Policy Optimization

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

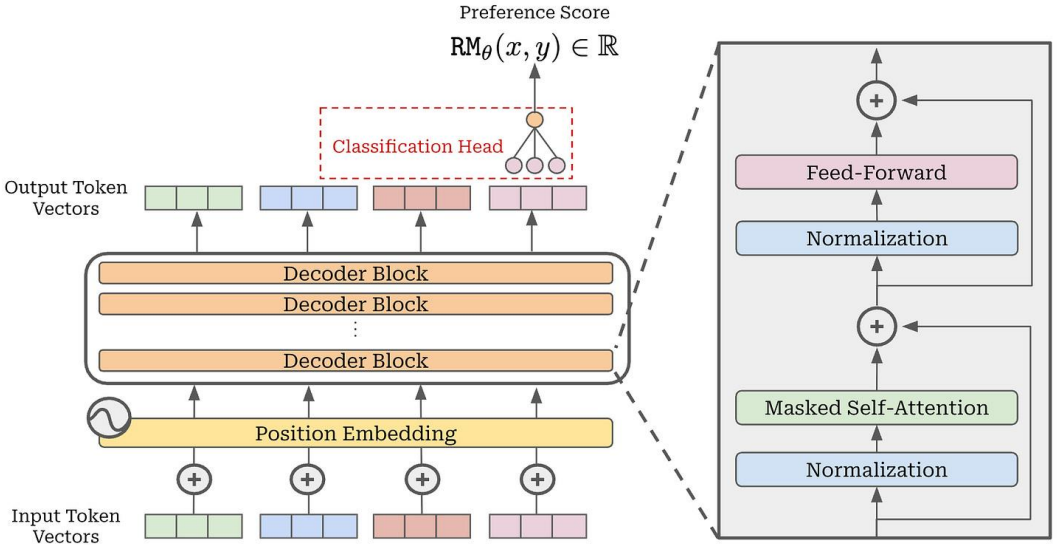
policy model $\pi_\theta \leftarrow \pi_{\theta_{init}}$
for iteration = 1, . . . , I **do**
  reference model $\pi_{ref} \leftarrow \pi_\theta$
  for step = 1, . . . , M **do**
    Sample a batch $\mathcal{D}_b$ from $\mathcal{D}$
    Update the old policy model $\pi_{\theta_{old}} \leftarrow \pi_\theta$
    Sample G outputs $\{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(\cdot \mid q)$ for each question $q \in \mathcal{D}_b$
    Compute rewards $\{r_i\}_{i=1}^{G}$ for each sampled output o_i by running r_φ
    Compute $\hat{A}_{i,t}$ 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

DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. Zhihong Shao, et al (2024). CoRR.



Reward Model

Specialized LLM finetuned to predict human preference score for a given prompt and candidate completion



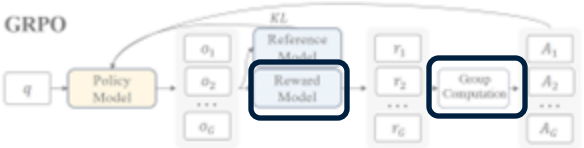
Loss function for RM with parameters θ

Difference between chosen and rejected score

$$\mathcal{L}(\theta) = -\log\left(\sigma\left(\text{RM}_\theta(x, y_c) - \text{RM}_\theta(x, y_r)\right)\right)$$

Logistic / Sigmoid function

Deep (Learning) Focus - Group Relative Policy Optimization (GRPO). Cameron R. Wolfe (2025). Group Relative Policy Optimization (GRPO)



Rewards and Group Relative Advantages

PPO Per-token rewards. π_{ref} is the reference model (initial SFT model)

$$r_t = r_\phi(q, a_{\leq t}) - \beta \log \frac{\pi_\theta(a_t \mid q, a_{<t})}{\pi_{ref}(a_t \mid q, a_{<t})}$$

GRPO Removing the value network

Estimates baseline advantages by normalizing a group (batch) of rewards obtained from sampling multiple different outputs from the reference policy:

$$\widehat{A_{i,t}} = \widetilde{r_i} = \frac{r_i - \text{mean}(r)}{\text{std}(r)}$$

t-th token in i-th output

Prompt (or question) sampled from dataset: $x \sim \mathcal{D}$

Group of sampled outputs (from old policy): $\mathbf{y} = \{y_1, y_2, \ldots, y_G\}$

Outcome reward for each output: $\mathbf{r} = \{r_1, r_2, \ldots, r_G\}$

Deep (Learning) Focus - Group Relative Policy Optimization (GRPO). Cameron R. Wolfe (2025). Group Relative Policy Optimization (GRPO)

Outcome Supervision RL vs Process Supervision RL

Sample a group of G outputs from the old policy $\pi_{\theta_{old}}$: $\{o_1, o_2, \ldots, o_G\}$

Outcome Supervision (OS) RL:

$$\tilde{r}_i = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}, \quad \text{where } \mathbf{r} = (r_1, \ldots, r_G)$$

$$\hat{A}_{i,t} = \tilde{r}_i, \quad \forall t \in \{1, \ldots, T_i\}$$

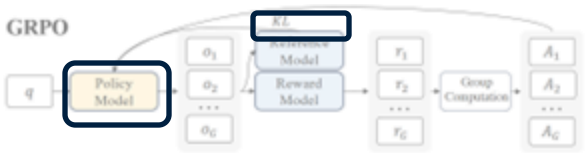
Process Supervision (PS) RL:

For each output o_i with K_i steps, let $index_i(j)$ be the token index of the end token of step j in output i:

$$\mathbf{R} = \{\{r_1^{index(1)}, \cdots, r_1^{index(K_1)}\}, \cdots, \{r_G^{index(1)}, \cdots, r_G^{index(K_G)}\}\}$$

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

Empirical study shows that GRPO+PS generally performs better



KL Estimator

- Unbiased KL estimator and low variance, always positive (r > 0)
- Adds KL divergence term directly in loss to simplifies advantage calculation $\widehat{A_{i,t}}$
- Reduce complex computation of per-token rewards (more frequent than the objective)

$$\mathbb{D}_{\mathbb{KL}}[\pi_\theta \mid \pi_{\text{ref}}] = \left\{\frac{\pi_{\text{ref}}(o_{i,t} \mid q, o_{i,<t})}{\pi_\theta(o_{i,t} \mid q, o_{i,<t})} - 1\right\} - \log\left(\frac{\pi_{\text{ref}}(o_{i,t} \mid q, o_{i,<t})}{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}\right)$$

Standard KL Estimator for LLMs (a.k.a. k_1)

Reference Distribution

$$\mathrm{KL}(q, p) \approx \log\left(\frac{q(x)}{p(x)}\right) = \log q(x) - \log p(x)$$

Policy Distribution

DeepSeekMath KL Estimator (a.k.a. k_3)

$$\mathrm{KL}(q, p) \approx (r - 1) - \log r, \text{ where } r = \frac{q(x)}{p(x)}$$

GRPO does not always perform multiple policy updates per batch of data. If we only perform a single policy update per batch, we have:

$$\min\left(\frac{\pi_\theta(a_{i,t}|s_{i,t})}{\pi_{\texttt{old}}(a_{i,t}|s_{i,t})}A_{i,t}, \texttt{clip}\left(\frac{\pi_\theta(a_{i,t}|s_{i,t})}{\pi_{\texttt{old}}(a_{i,t}|s_{i,t})}, 1-\epsilon, 1+\epsilon\right)A_{i,t}\right)$$

$$= \frac{\pi_\theta(a_{i,t}|s_{i,t})}{[\pi_\theta(a_{i,t}|s_{i,t})]_\nabla}A_{i,t} \quad \text{because} \quad \pi_{\texttt{old}} = \pi_\theta$$

Stop Gradient

Deep (Learning) Focus - Group Relative Policy Optimization (GRPO). Cameron R. Wolfe (2025). Group Relative Policy Optimization (GRPO), Approximating KL Divergence John Schulman

Iterative Supervision RL with GRPO

- Generate new training sets for the reward model based on the sampling results from the policy model
- Training reward model using replay mechanism that incorporates 10% of historical data
- Update the reference model as policy model, training new policy model with new reward model

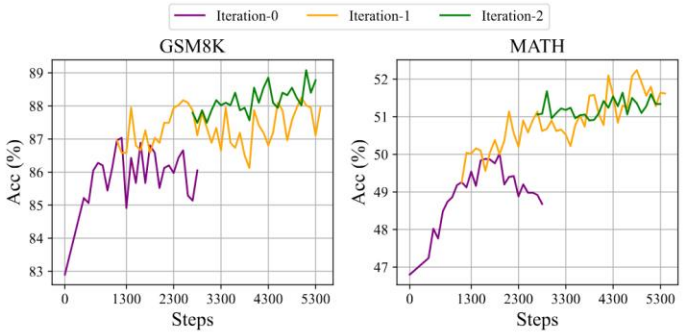


Figure 6 | Performance of iterative reinforcement learning with DeepSeekMath-Instruct 7B on two benchmarks.

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 $\pi_\theta \leftarrow \pi_{\theta_{\text{init}}}$
for iteration = 1, . . . , I **do**
  reference model $\pi_{ref} \leftarrow \pi_\theta$
  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 $\{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(\cdot \mid q)$ for each question $q \in \mathcal{D}_b$
    Compute rewards $\{r_i\}_{i=1}^{G}$ for each sampled output o_i by running r_φ
    Compute $\hat{A}_{i,t}$ 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

Memory Efficiency

PPO memory consumption during training:

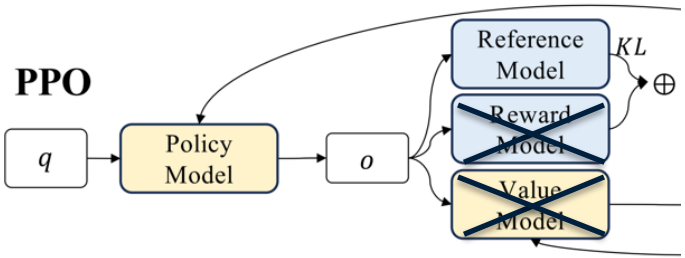
Training: Policy Model + Critic Model

Inference: Reward Model + Reference Policy Model

Assume half precision (bf16/fp16)

During Inference: ~2GB per 1B (model size) to store model parameters

During training: ~16GB per 1B (model size) to store, cache activations during forward pass, gradients during back pass, and gradient history for Adam



With RLVR can further replace neural reward model with heuristic verifier

GRPO - Replace critic with relative group advantages

Results and Discussion

Training and Evaluating DeepSeekMath-RL

RL based on DeepSeek-Instruct 7B.

Training Data: CoT GSM8K and MATH from SFT data, 144K questions.

Initial reward model: DeepSeekMath-Base 7B

GRPO: Sample 64 outputs, max length is set to 1024, batch size 1024

Performance surpasses all open-source models in 7B to 70B

DeepSeekMath-RL outperforms SFT model across all evaluation metrics

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%
Open-Source Model					
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%**

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%**

Lesson Learnt in Pre-Training

- Code training Benefits Mathematical Reasoning – improves models' ability to do math reasoning, w/wo tool use. Experiment with DeepSeek-LLM 1.3B.
- One-stage mixed training is worse due to limited model scale, lacks the capacity to fully assimilate both code and math data.
- ArXiv Papers seem ineffective in improving mathematical reasoning

Training Setting	Training Tokens			w/o Tool Use			w/ Tool Use	
	General	Code	Math	GSM8K	MATH	CMATH	GSM8K+Python	MATH+Python
No Continual Training	–	–	–	2.9%	3.0%	12.3%	2.7%	2.3%
Two-Stage Training								
Stage 1: General Training	400B	–	–	2.9%	3.2%	14.8%	3.3%	2.3%
Stage 2: Math Training	–	–	150B	19.1%	14.4%	37.2%	14.3%	6.7%
Stage 1: Code Training	–	400B	–	5.9%	3.6%	19.9%	12.4%	10.0%
Stage 2: Math Training	–	–	150B	**21.9%**	**15.3%**	**39.7%**	17.4%	9.4%
One-Stage Training								
Math Training	–	–	150B	20.5%	13.1%	37.6%	11.4%	6.5%
Code & Math Mixed Training	–	400B	150B	17.6%	12.1%	36.3%	**19.7%**	**13.5%**

Insights of Reinforcement Learning

Unified Paradigm to analyze different training methods (SFT, RFT, DPO, PPO, GRPO)

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

D: Online sampling > Offline sampling
GC: Model > Rule

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

Table 10 | The data source and gradient coefficient of different methods. P_{sft} denotes the data distribution of supervised fine-tuning datasets. $\pi_{\theta_{sft}}$ and π_θ denote the supervised fine-tuned model and the real-time policy model during the online training process, respectively.

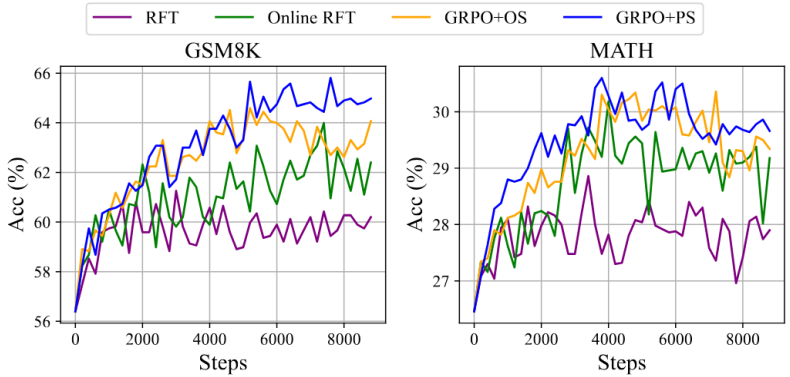


Figure 5 | Performance of the DeepSeekMath-Instruct 1.3B model, which was further trained using various methods, on two benchmarks.

Evaluation Metrics: Maj@K and Pass@K

- RL enhances performance by rendering the output distribution more robust
- Boosting the correct response from Top K, rather than the enhancement of fundamental capabilities

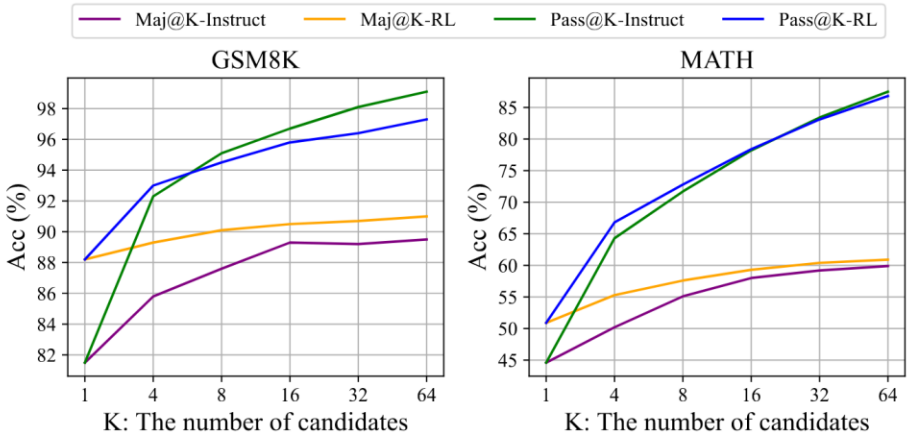


Figure 7 | The Maj@K and Pass@K of SFT and RL DeepSeekMath 7B on GSM8K and MATH (temperature 0.7). It was noted that RL enhances Maj@K but not Pass@K.

Maj@K – sample K solutions, extract their **final answers**, take the **most frequent (majority/plurality) answer**, and check if that voted answer is correct.
Pass@K – problem is counted as solved if at least one of the K sampled outputs is correct.

GRPO Done Right – Tricks and Improvements
Dr. GRPO, Routing Replay, GSPO

Limitations and Problems

Limitations of vanilla GRPO

- Noise and instability during training process
- Excessive response lengths, especially in incorrect answers
- Collapse of LLM's entropy (reduced exploration)
- Poor sample efficiency and slow learning

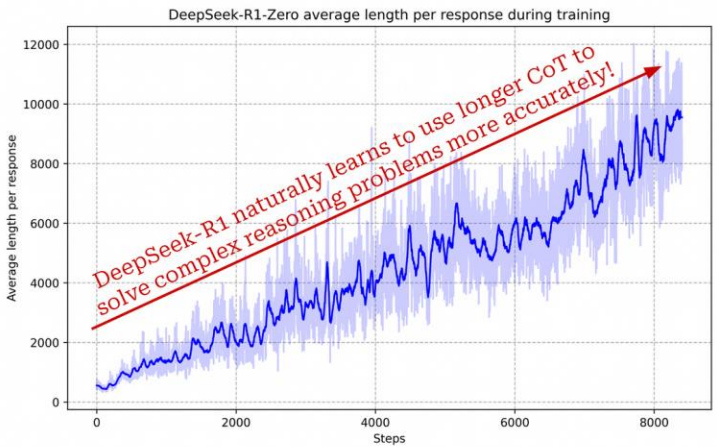


Figure 3 | The average response length of DeepSeek-R1-Zero on the training set during the RL process. DeepSeek-R1-Zero naturally learns to solve reasoning tasks with more thinking time.

Deep (Learning) Focus - GRPO++: Tricks for Making RL Actually Work. Cameron R. Wolfe (2025). https://cameronrwolfe.substack.com/p/grpo-tricks

Modification from the GRPO objectives

Response-level length bias: GRPO objective prefer shorter but better output; also penalize more on shorter but worse output.

Question-level difficulty bias: Questions with lower SD are given higher weights during policy update. Question-level normalization results in varying weights in easy and hard questions.

GRPO

$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|\mathbf{o}_i|}\sum_{t=1}^{|\mathbf{o}_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}\hat{A}_{i,t},\text{clip}\left(\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})},1-\epsilon,1+\epsilon\right)\hat{A}_{i,t}\right]\right\},$$

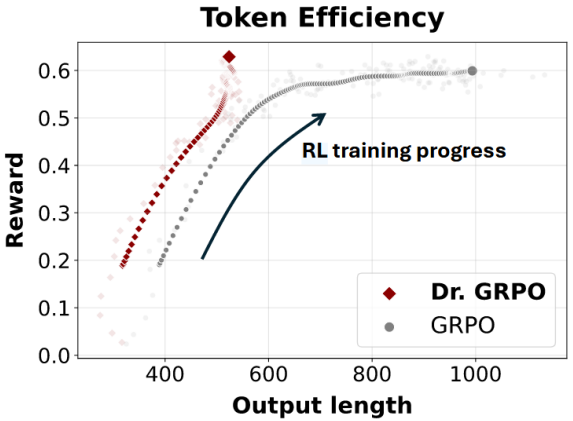
$$\text{where } \hat{A}_{i,t}=\frac{R(\mathbf{q},\mathbf{o}_i)-\text{mean}(\{R(\mathbf{q},\mathbf{o}_1),\ldots,R(\mathbf{q},\mathbf{o}_G)\})}{\text{std}(\{R(\mathbf{q},\mathbf{o}_1),\ldots,R(\mathbf{q},\mathbf{o}_G)\})}.$$

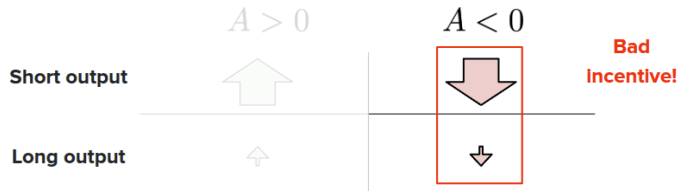
Dr. GRPO

GRPO Done Right (without bias)

$$\frac{1}{G}\sum_{i=1}^{G}\sum_{t=1}^{|\mathbf{o}_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}\hat{A}_{i,t},\text{clip}\left(\frac{\pi_\theta(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|\mathbf{q},\mathbf{o}_{i,<t})},1-\epsilon,1+\epsilon\right)\hat{A}_{i,t}\right]\right\},$$

$$\text{where } \hat{A}_{i,t}=R(\mathbf{q},\mathbf{o}_i)-\text{mean}(\{R(\mathbf{q},\mathbf{o}_1),\ldots,R(\mathbf{q},\mathbf{o}_G)\}).$$





Understanding R1-Zero-Like Training: A Critical Perspective. Zichen Liu, et al (2025). COLM 2025.

Routing Replay and Group Sequence Policy Optimization

Routing Replay training strategy: cache activated experts in old policy and replay these routing modes (MoE model) so they policy ratio compute with same activated network

Based on this straightforward observation, we propose the **Group Sequence Policy Optimization (GSPO)** algorithm. GSPO employs the following sequence-level optimization objective:

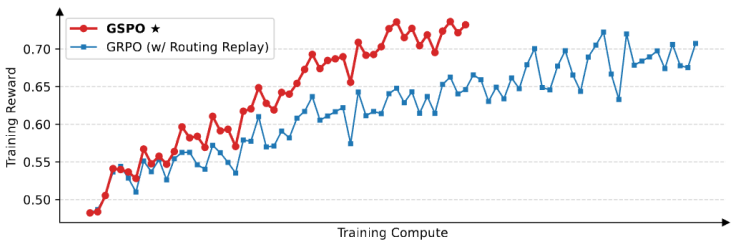
$$\mathcal{J}_{\text{GSPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|x)}\left[\frac{1}{G}\sum_{i=1}^{G}\min\left(s_i(\theta)\widehat{A}_i,\ \text{clip}\left(s_i(\theta),1-\varepsilon,1+\varepsilon\right)\widehat{A}_i\right)\right], \tag{5}$$

where we adopt the group-based advantage estimation:

$$\widehat{A}_i = \frac{r(x,y_i) - \text{mean}\left(\{r(x,y_i)\}_{i=1}^{G}\right)}{\text{std}\left(\{r(x,y_i)\}_{i=1}^{G}\right)}, \tag{6}$$

and define the importance ratio $s_i(\theta)$ based on sequence likelihood (Zheng et al., 2023):

$$s_i(\theta) = \left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{\text{old}}}(y_i|x)}\right)^{\frac{1}{|y_i|}} = \exp\left(\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\log\frac{\pi_\theta(y_{i,t}|x,y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|x,y_{i,<t})}\right).$$



Group Sequence Policy Optimization. Chujie Zheng, et al (2025).

More Effective RL

Data Source:

- RL pipeline on out-of-distribution question prompts, in conjunction with advanced sampling (decoding strategies)
- Efficient inference techniques that determine the exploration efficiency of policy models

Algorithm:

- All methods fully trust the signal of the reward function, regardless of signal reliability
- Annotated datasets like PRM800K contain approx. 20% of incorrectly annotations
- RL algorithm robust against noisy reward signals, e.g., weak-to-strong alignment

Reward Function:

- Enhance the generalization ability of the reward model to handle out-of-distribution questions and advanced decoding outputs
- Reflect the uncertainty of the reward model
- Efficiently build high-quality process reward models that provide fine-grained training signals for the reasoning process

Conclusion

- DeepSeekMath achieved strong performance on the benchmarks and outperformed all existing open-source models
- Group Relative Policy Optimization (GRPO) improves reasoning performance with lower memory cost

Questions

- Does the dataset contain bias toward certain math topics?
- How well the performance might be with different math domains beyond the benchmarks from the paper?

Possible Future Work

- Can scaling the model to larger sizes
- Can expand the dataset to cover more mathematical domains

Reference

Publications:

Policy Gradient Methods for Reinforcement Learning with Function Approximation. Richard Sutton, et al. (1999). NeuralPS.

Proximal Policy Optimization Algorithms. John Schulman, et al (2017).

DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. Zhihong Shao, et al (2024). CoRR.

Understanding R1-Zero-Like Training: A Critical Perspective. Zichen Liu, et al (2025). COLM 2025.

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. DeepSeek-AI, et al (2025).

Online Resources:

Policy Gradient Methods in Reinforcement Learning. Vizuara AI (2025). Policy Gradient Methods in Reinforcement Learning

Reinforcement Learning from Human Feedback. Nathan Lambert (2026). RLHF Book by Nathan Lambert

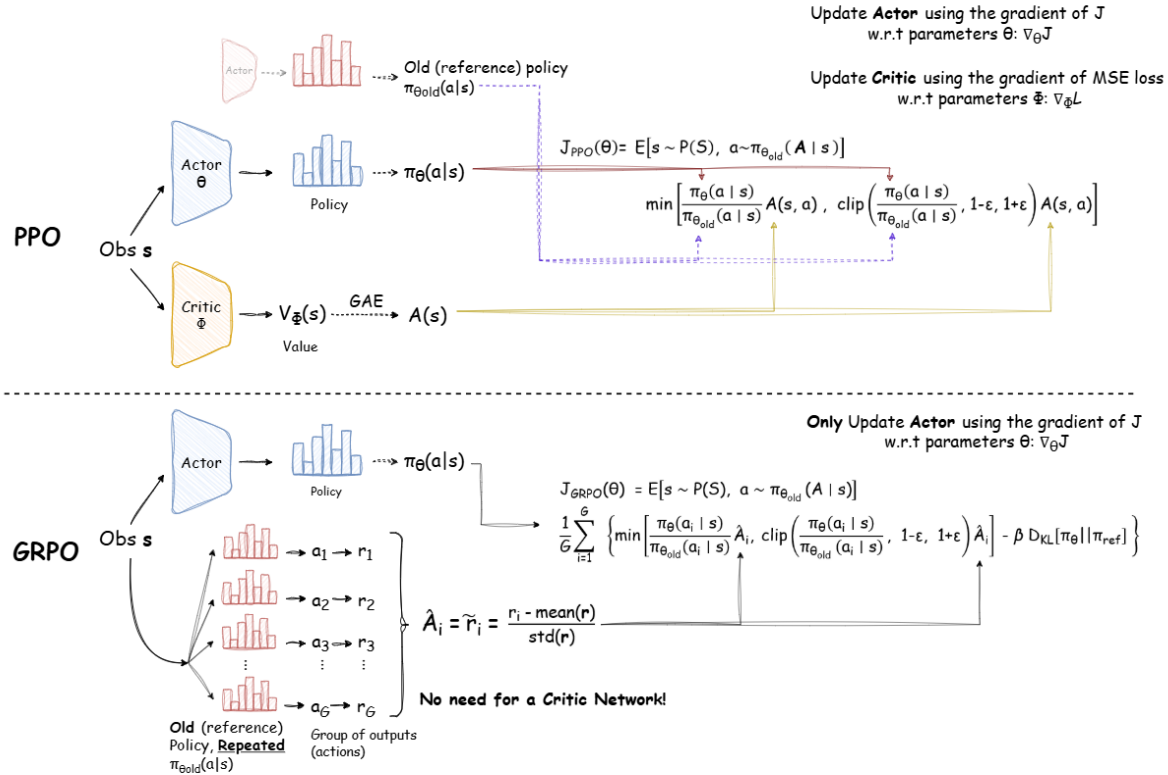
Deep (Learning) Focus - Group Relative Policy Optimization (GRPO). Cameron R. Wolfe (2025). https://cameronrwolfe.substack.com/p/grpo

Deep (Learning) Focus - PPO for LLMs: A Guide for Normal People. Cameron R. Wolfe (2025). https://cameronrwolfe.substack.com/p/ppo-llm

Deep (Learning) Focus - GRPO++: Tricks for Making RL Actually Work. Cameron R. Wolfe (2025). https://cameronrwolfe.substack.com/p/grpo-tricks

Group Relative Policy Optimization (GRPO) Illustrated Breakdown. Ebrahim Pichka (2025). Group Relative Policy Optimization (GRPO) Illustrated Breakdown | Ebrahim Pichka

Appendix: From PPO to GRPO



Group Relative Policy Optimization (GRPO) Illustrated Breakdown. Ebrahim Pichka (2025). Group Relative Policy Optimization (GRPO) Illustrated Breakdown | Ebrahim Pichka

Data Source: question in SFT dataset with outputs sampled from SFT model. Reward Function: Rule (whether the answer is correct or not). Gradient Coefficient:

$$GC_{RFT}(q, o, t) = \mathbb{I}(o) = \begin{cases} 1 & \text{the answer of o is correct} \\ 0 & \text{the answer of o is incorrect} \end{cases} \tag{10}$$

Data Source: question in SFT dataset with outputs sampled from policy model. Reward Function: reward model. Gradient Coefficient:

$$GC_{PPO}(q, o, t, \pi_{\theta_{rm}}) = A_t, \tag{18}$$

where A_t is the advantage, which is computed by applying Generalized Advantage Estimation (GAE) (Schulman et al., 2015), based on the rewards $\{r_{\geq t}\}$ and a learned value function V_ψ.

Data Source: question in SFT dataset with outputs sampled from policy model. Reward Function: reward model. Gradient Coefficient:

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

where $\hat{A}_{i,t}$ is computed based on the group reward scores.

THE UNIVERSITY OF BRITISH COLUMBIA