

DeepSeek: Decoding the Technical Insights Behind Model Evolution

Shuo Yang

Technical Overview

1. DeepSeekMoE
2. From PPO to GRPO
3. From MHA, GQA to MLA
4. MTP
5. NSA

Model Evolution

1. DeepSeek LLM
2. DeepSeek-V2
3. DeepSeek-V3
4. DeepSeek-R1



DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models

Damai Dai^{*1,2}, Chengqi Deng¹, Chenggang Zhao^{*1,3}, R.X. Xu¹, Huazuo Gao¹, Deli Chen¹, Jiashi Li¹, Wangding Zeng¹, Xingkai Yu^{*1,4}, Y. Wu¹, Zhenda Xie¹, YK. Li¹, Pannan Huang¹, Fuli Luo¹, Chong Ruan¹,
Zhifang Sui², Wenfeng Liang¹

¹DeepSeek-AI

²National Key Laboratory for Multimedia Information Processing, Peking University

³Institute for Interdisciplinary Information Sciences, Tsinghua University

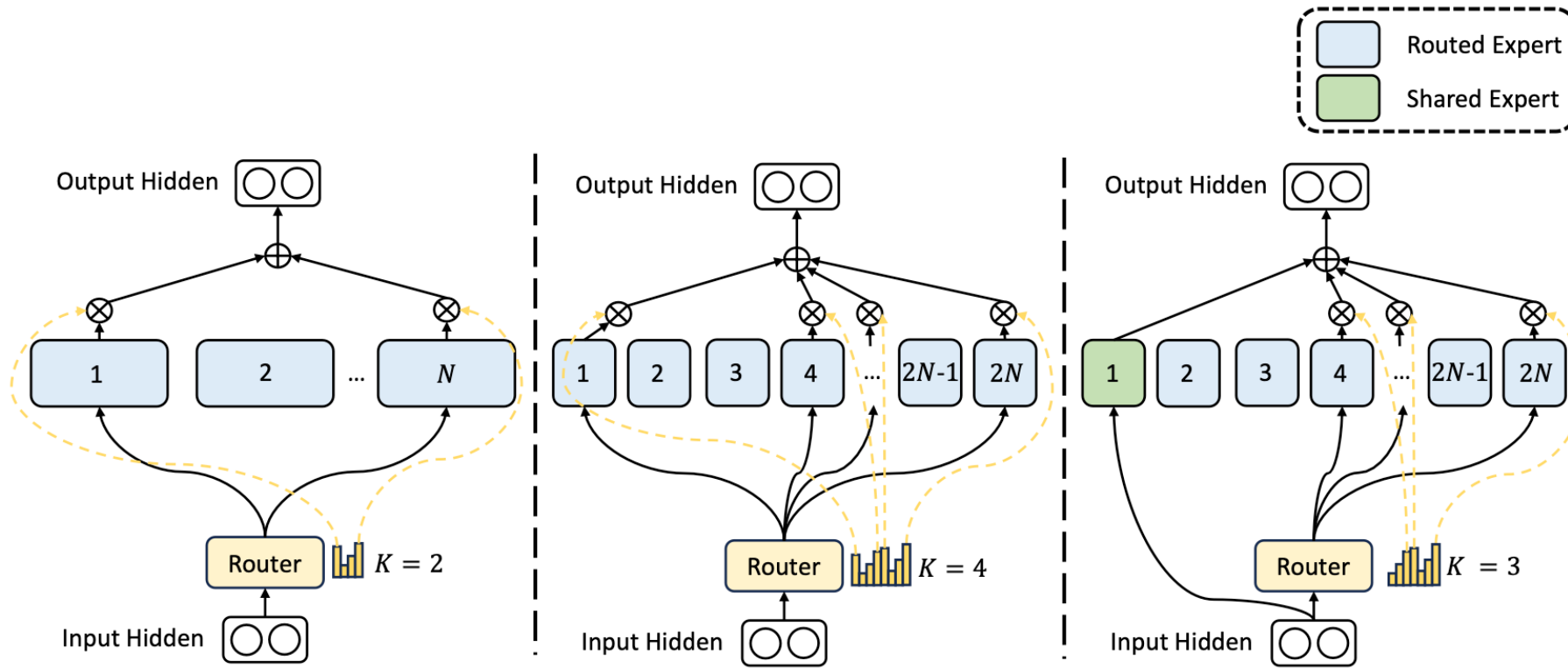
⁴National Key Laboratory for Novel Software Technology, Nanjing University

{daidamai, szf}@pku.edu.cn, {wenfeng.liang}@deepseek.com

<https://github.com/deepseek-ai/DeepSeek-MoE>

- Knowledge Mixture
- Knowledge Redundancy

- Fine-Grained Expert Segmentation
- Shared Expert Isolation



(a) Conventional Top-2 Routing $\longrightarrow$ **(b) + Fine-grained Expert Segmentation** $\longrightarrow$ **(c) + Shared Expert Isolation (DeepSeekMoE)**

$$\mathbf{h}_t^l = \sum_{i=1}^N \left(g_{i,t} \text{FFN}_i(\mathbf{u}_t^l) \right) + \mathbf{u}_t^l,$$

$$g_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} \in \text{Topk}(\{s_{j,t} | 1 \leq j \leq N\}, K), \\ 0, & \text{otherwise}, \end{cases}$$

$$s_{i,t} = \text{Softmax}_i(\mathbf{u}_t^{lT} \mathbf{e}_i^l),$$

$$\mathbf{h}_t^l = \sum_{i=1}^{mN} \left(g_{i,t} \text{FFN}_i(\mathbf{u}_t^l) \right) + \mathbf{u}_t^l,$$

$$g_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} \in \text{Topk}(\{s_{j,t} | 1 \leq j \leq mN\}, mK), \\ 0, & \text{otherwise}, \end{cases}$$

$$s_{i,t} = \text{Softmax}_i(\mathbf{u}_t^{lT} \mathbf{e}_i^l),$$

$$\mathbf{h}_t^l = \sum_{i=1}^{K_s} \text{FFN}_i(\mathbf{u}_t^l) + \sum_{i=K_s+1}^{mN} \left(g_{i,t} \text{FFN}_i(\mathbf{u}_t^l) \right) + \mathbf{u}_t^l,$$

$$g_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} \in \text{Topk}(\{s_{j,t} | K_s + 1 \leq j \leq mN\}, mK - K_s), \\ 0, & \text{otherwise}, \end{cases}$$

$$s_{i,t} = \text{Softmax}_i(\mathbf{u}_t^{lT} \mathbf{e}_i^l).$$

Expert-Level Balance Loss. In order to mitigate the risk of routing collapse, we also employ an expert-level balance loss. The computation of the balance loss is as follows:

$$\mathcal{L}_{\text{ExpBal}} = \alpha_1 \sum_{i=1}^{N'} f_i P_i, \quad (12)$$

$$f_i = \frac{N'}{K'T} \sum_{t=1}^T \mathbb{1}(\text{Token } t \text{ selects Expert } i), \quad (13)$$

$$P_i = \frac{1}{T} \sum_{t=1}^T s_{i,t}, \quad (14)$$

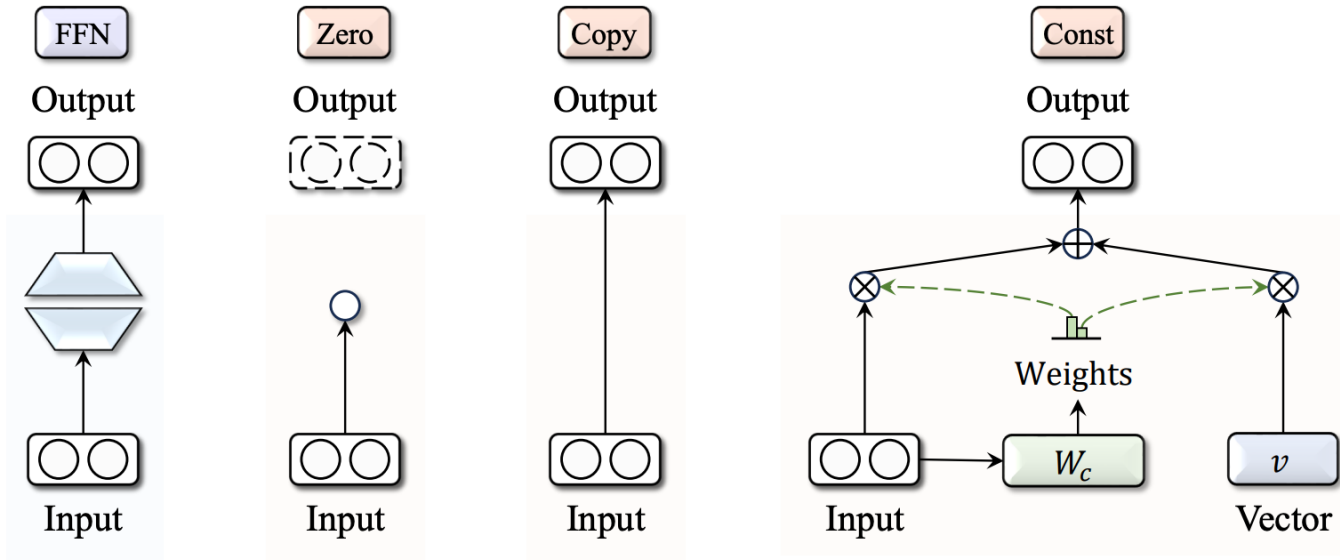
where α_1 is a hyper-parameter called expert-level balance factor, N' is equal to $(mN - K_s)$ and K' is equal to $(mK - K_s)$ for brevity. $\mathbb{1}(\cdot)$ denotes the indicator function.

Device-Level Balance Loss. In addition to the expert-level balance loss, we introduce a device-level balance loss. When aiming to alleviate computation bottlenecks, it becomes unnecessary to enforce strict balance constraints at the expert level, because excessive constraints on load balance will compromise model performance. Instead, our primary objective is to ensure balanced computation across the devices. If we partition all routed experts into D groups $\{\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_D\}$, and deploy each group on a single device, the device-level balance loss is computed as follows:

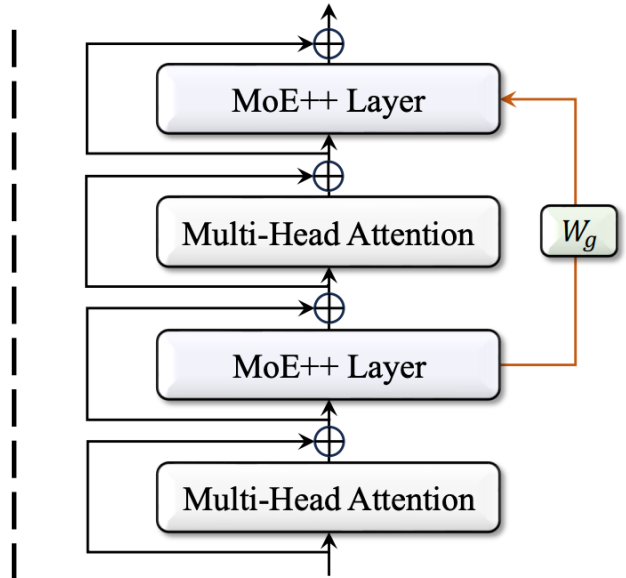
$$\mathcal{L}_{\text{DevBal}} = \alpha_2 \sum_{i=1}^D f'_i P'_i, \quad (15)$$

$$f'_i = \frac{1}{|\mathcal{E}_i|} \sum_{j \in \mathcal{E}_i} f_j, \quad (16)$$

$$P'_i = \sum_{j \in \mathcal{E}_i} P_j, \quad (17)$$

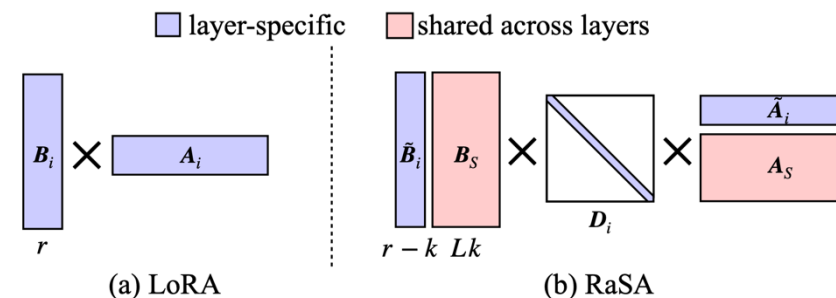


(a) Vanilla FFN expert and zero-computation experts



(b) Gating residuals

Future Work Directions



- ❑ **Shared + Fine-Grained in more scenes (Resnet / Skipping)**
 - ❑ Continuous Learning. (**Fine-Grained and Balance**) (AM-LoRA++)
- ❑ **Dynamic Optimization of Splitting Factors in Combination with RL**
 - ❑ MoE++
- ❑ **Modal Expert Extension**
 - ❑ Designing specialized experts for modalities such as image and speech, and constructing a unified multimodal MoE architecture.



Metric	# Shot	Dense	Hash Layer	Switch	GShard	DeepSeekMoE
# Total Params	N/A	0.2B	2.0B	2.0B	2.0B	2.0B
# Activated Params	N/A	0.2B	0.2B	0.2B	0.3B	0.3B
FLOPs per 2K Tokens	N/A	2.9T	2.9T	2.9T	4.3T	4.3T
# Training Tokens	N/A	100B	100B	100B	100B	100B
Pile (Loss)	N/A	2.060	1.932	1.881	1.867	1.808
HellaSwag (Acc.)	0-shot	38.8	46.2	49.1	50.5	54.8
PIQA (Acc.)	0-shot	66.8	68.4	70.5	70.6	72.3
ARC-easy (Acc.)	0-shot	41.0	45.3	45.9	43.9	49.4
ARC-challenge (Acc.)	0-shot	26.0	28.2	30.2	31.6	34.3
RACE-middle (Acc.)	5-shot	38.8	38.8	43.6	42.1	44.0
RACE-high (Acc.)	5-shot	29.0	30.0	30.9	30.4	31.7
HumanEval (Pass@1)	0-shot	0.0	1.2	2.4	3.7	4.9
MBPP (Pass@1)	3-shot	0.2	0.6	0.4	0.2	2.2
TriviaQA (EM)	5-shot	4.9	6.5	8.9	10.2	16.6
NaturalQuestions (EM)	5-shot	1.4	1.4	2.5	3.2	5.7

Table 1 | Evaluation results for validation experiments. **Bold** font indicates the best. Compared with other MoE architectures, DeepSeekMoE exhibits a substantial performance advantage.

基准模型 在验证实验中，包括DeepSeekMoE在内，我们比较了五个模型。Dense表示一个标准的稠密Transformer语言模型，具有0.2B总参数。Hash Layer (Roller et al., 2021) 是一个基于top-1哈希路由的MoE架构，具有2.0B总参数和0.2B激活参数，与稠密基准模型对齐。Switch Transformer (Fedus et al., 2021) 是另一个基于top-1可学习路由的著名MoE架构，总参数和激活参数与Hash Layer相同。GShard (Lepikhin et al., 2021) 采用top-2可学习路由策略，具有2.0B总参数和0.3B激活参数，因为与top-1路由方法相比，激活了一个额外的专家。DeepSeekMoE具有1个共享专家和63个路由专家，每个专家的大小是标准FFN的0.25倍。包括DeepSeekMoE在内，所有比较的模型共享相同的训练语料库和训练超参数。所有比较的MoE模型具有相同数量的总参数，且GShard与DeepSeekMoE具有相同数量的激活参数。

Metric	# Shot	GShard×1.5	Dense×16	DeepSeekMoE
Relative Expert Size	N/A	1.5	1	0.25
# Experts	N/A	0 + 16	16 + 0	1 + 63
# Activated Experts	N/A	0 + 2	16 + 0	1 + 7
# Total Expert Params	N/A	2.83B	1.89B	1.89B
# Activated Expert Params	N/A	0.35B	1.89B	0.24B
FLOPs per 2K Tokens	N/A	5.8T	24.6T	4.3T
# Training Tokens	N/A	100B	100B	100B
Pile (Loss)	N/A	1.808	1.806	1.808
HellaSwag (Acc.)	0-shot	54.4	55.1	54.8
PIQA (Acc.)	0-shot	71.1	71.9	72.3
ARC-easy (Acc.)	0-shot	47.3	51.9	49.4
ARC-challenge (Acc.)	0-shot	34.1	33.8	34.3
RACE-middle (Acc.)	5-shot	46.4	46.3	44.0
RACE-high (Acc.)	5-shot	32.4	33.0	31.7
HumanEval (Pass@1)	0-shot	3.0	4.3	4.9
MBPP (Pass@1)	3-shot	2.6	2.2	2.2
TriviaQA (EM)	5-shot	15.7	16.5	16.6
NaturalQuestions (EM)	5-shot	4.7	6.3	5.7

Table 2 | Comparisons among DeepSeekMoE, larger GShard models, and larger dense models. In the line of “# Experts”, $a + b$ denotes a shared experts and b routed experts. In the line of “# Activated Experts”, $a + b$ denotes a activated shared experts and b activated routed experts. DeepSeekMoE achieves comparable performance with a GShard model containing 1.5 times expert parameters and computation. In addition, DeepSeekMoE nearly approaches the performance of a dense model with 16 times FFN parameters, which sets the upper bound for MoE models in terms of the model capacity.

- 1. DeepSeekMoE 在多个任务上展示了与 GShard×1.5 和 Dense×16 相当的性能。
- 2. 其设计显示了在不显著增加参数的情况下，可以实现与更大模型相媲美的效果

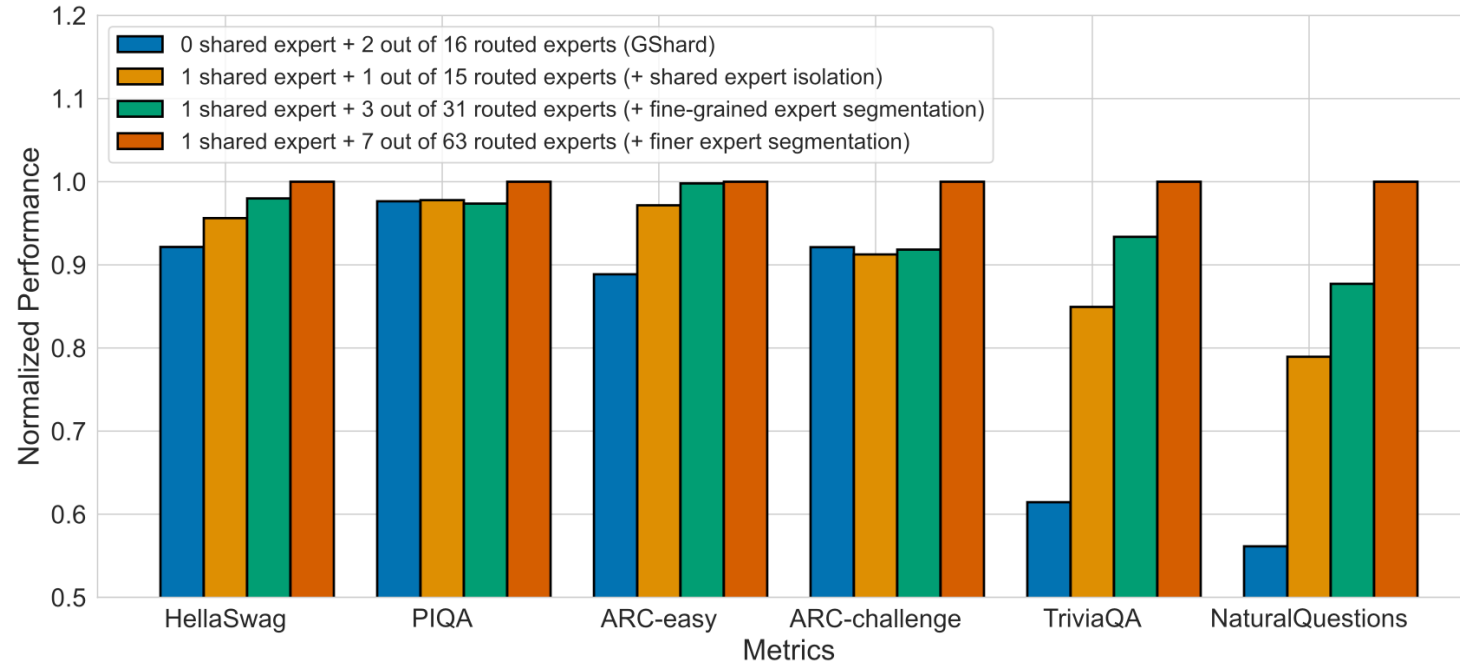
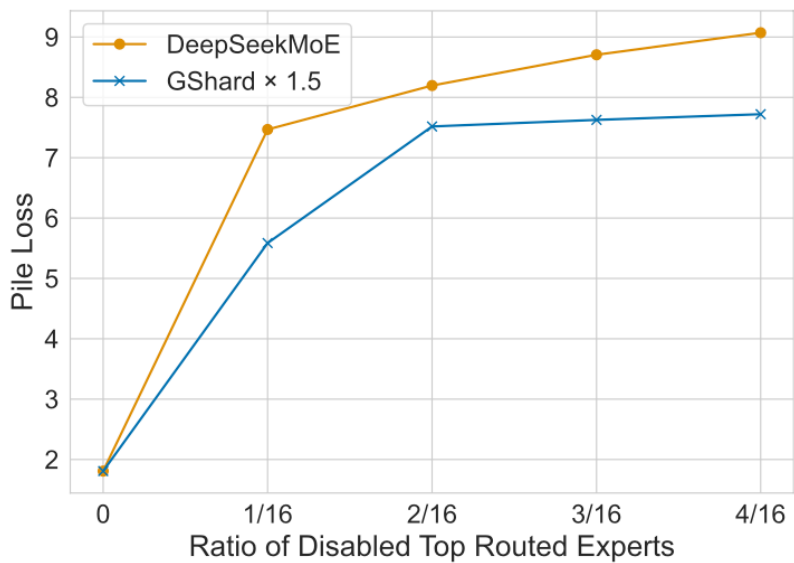
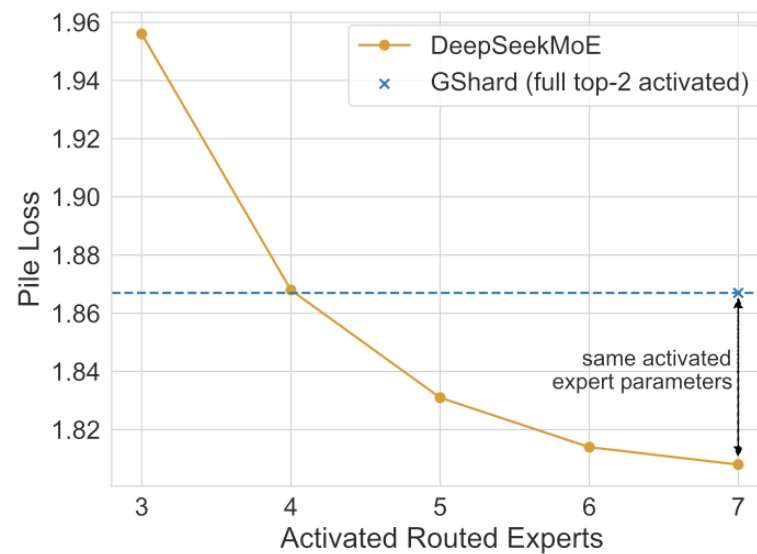


Figure 3 | Ablation studies for DeepSeekMoE. The performance is normalized by the best performance for clarity in presentation. All compared models have the same number of parameters and activated parameters. We can find that fine-grained expert segmentation and shared expert isolation both contribute to stronger overall performance.



GShard $\times 1.5$ 相比，DeepSeekMoE对禁用顶级路由专家的比例更为敏感，表明DeepSeekMoE在路由专家之间的冗余度较低。



即使只激活4个路由专家，DeepSeekMoE的Pile损失与GShard相当，展示了它在使用较少专家的情况下仍能高效工作。



Metric	# Shot	LLaMA2 SFT 7B	DeepSeek Chat 7B	DeepSeekMoE Chat 16B
# Total Params	N/A	6.7B	6.9B	16.4B
# Activated Params	N/A	6.7B	6.9B	2.8B
FLOPs per 4K Tokens	N/A	187.9T	183.5T	74.4T
HellaSwag (Acc.)	0-shot	67.9	71.0	72.2
PIQA (Acc.)	0-shot	76.9	78.4	79.7
ARC-easy (Acc.)	0-shot	69.7	70.2	69.9
ARC-challenge (Acc.)	0-shot	50.8	50.2	50.0
BBH (EM)	3-shot	39.3	43.1	42.2
RACE-middle (Acc.)	5-shot	63.9	66.1	64.8
RACE-high (Acc.)	5-shot	49.6	50.8	50.6
DROP (EM)	1-shot	40.0	41.7	33.8
GSM8K (EM)	0-shot	63.4	62.6	62.2
MATH (EM)	4-shot	13.5	14.7	15.2
HumanEval (Pass@1)	0-shot	35.4	45.1	45.7
MBPP (Pass@1)	3-shot	27.8	39.0	46.2
TriviaQA (EM)	5-shot	60.1	59.5	63.3
NaturalQuestions (EM)	0-shot	35.2	32.7	35.1
MMLU (Acc.)	0-shot	50.0	49.7	47.2
WinoGrande (Acc.)	0-shot	65.1	68.4	69.0
CLUEWSC (EM)	5-shot	48.4	66.2	68.2
CEval (Acc.)	0-shot	35.1	44.7	40.0
CMMLU (Acc.)	0-shot	36.9	51.2	49.3

Metric	# Shot	DeepSeek 67B (Dense)	GShard 137B	DeepSeekMoE 145B	DeepSeekMoE 142B (Half Activated)
# Total Params	N/A	67.4B	136.5B	144.6B	142.3B
# Activated Params	N/A	67.4B	21.6B	22.2B	12.2B
Relative Expert Size	N/A	N/A	1	0.125	0.125
# Experts	N/A	N/A	0 + 16	4 + 128	2 + 128
# Activated Experts	N/A	N/A	0 + 2	4 + 12	2 + 6
FLOPs per 4K Tokens	N/A	2057.5T	572.7T	585.6T	374.6T
# Training Tokens	N/A	245B	245B	245B	245B
Pile (Loss.)	N/A	1.905	1.961	1.876	1.888
HellaSwag (Acc.)	0-shot	74.8	72.0	75.8	74.9
PIQA (Acc.)	0-shot	79.8	77.6	80.7	80.2
ARC-easy (Acc.)	0-shot	69.0	64.0	69.7	67.9
ARC-challenge (Acc.)	0-shot	50.4	45.8	48.8	49.0
RACE-middle (Acc.)	5-shot	63.2	59.2	62.1	59.5
RACE-high (Acc.)	5-shot	46.9	43.5	45.5	42.6
DROP (EM)	1-shot	27.5	21.6	27.8	28.9
GSM8K (EM)	8-shot	11.8	6.4	12.2	13.8
MATH (EM)	4-shot	2.1	1.6	3.1	2.8
HumanEval (Pass@1)	0-shot	23.8	17.7	19.5	23.2
MBPP (Pass@1)	3-shot	33.6	27.6	33.2	32.0
TriviaQA (EM)	5-shot	57.2	52.5	61.1	59.8
NaturalQuestions (EM)	5-shot	22.6	19.0	25.0	23.5
MMLU (Acc.)	5-shot	45.1	26.3	39.4	37.5
WinoGrande (Acc.)	0-shot	70.7	67.6	71.9	70.8
CLUEWSC (EM)	5-shot	69.1	65.7	71.9	72.6
CEval (Acc.)	5-shot	40.3	26.2	37.1	32.8
CMMLU (Acc.)	5-shot	40.6	25.4	35.9	31.9
CHID (Acc.)	0-shot	88.5	86.9	90.3	88.3

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

Zhihong Shao^{1,2*†}, Peiyi Wang^{1,3*†}, Qihao Zhu^{1,3*†}, Runxin Xu¹, Junxiao Song¹
Xiao Bi¹, Haowei Zhang¹, Mingchuan Zhang¹, Y.K. Li¹, Y. Wu¹, Daya Guo^{1*}

¹DeepSeek-AI, ²Tsinghua University, ³Peking University

1. PPO is an Actor-Critic algorithm, but training the Critic/value model is costly
2. The value function estimation is difficult under sparse rewards
3. KL divergence calculation is complex

目标：训练一个Policy神经网络 π ，在所有的Trajectory中，得到Return的期望最大。

$$E(R(\tau))_{\tau \sim P_{\theta}(\tau)} = \sum_{\tau} R(\tau) P_{\theta}(\tau)$$

$$R(\tau^n) \rightarrow \sum_{t'=t}^{T_n} \gamma^{t'-t} r_{t'}^n = R_t^n$$

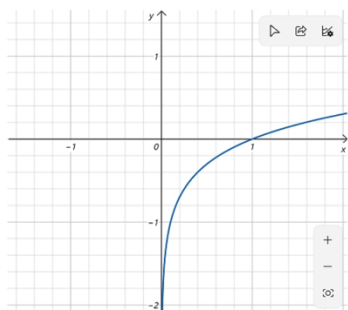
$$\nabla E(R(\tau))_{\tau \sim P_{\theta}(\tau)} = \nabla \sum_{\tau} R(\tau) P_{\theta}(\tau)$$

$$= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log P_{\theta}(a_n^t | s_n^t)$$

$$= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R_t^n - B(s_n^t)) \nabla \log P_{\theta}(a_n^t | s_n^t)$$

$$= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta}(s_n^t, a_n^t) \nabla \log P_{\theta}(a_n^t | s_n^t)$$

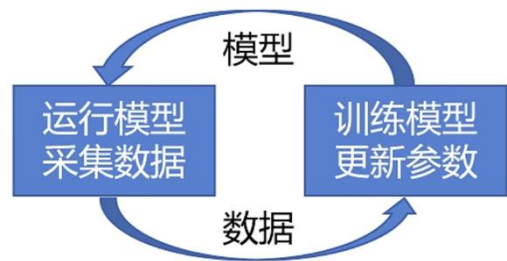
$$= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta}^{GAE}(s_n^t, a_n^t) \nabla \log P_{\theta}(a_n^t | s_n^t)$$



如果一个策略得到的return是大于0的，就增大该策略里所有state下取当前action的概率；
如果一个策略得到的return是小于0的，就减小该策略里所有state下取当前action的概率

$$\text{Loss} = -\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \log P_{\theta}(a_n^t | s_n^t)$$

$$A_{\theta}^{GAE}(s_t, a) = (1 - \lambda)(A_{\theta}^1 + \lambda * A_{\theta}^2 + \lambda^2 A_{\theta}^3 + \dots)$$



On Policy



Off Policy



$$\begin{aligned}
 E(f(x))_{x \sim p(x)} &= \sum_x f(x) * p(x) \\
 &= \sum_x f(x) * p(x) \frac{q(x)}{q(x)} \\
 &= \sum_x f(x) \frac{p(x)}{q(x)} * q(x) \\
 &= E(f(x) \frac{p(x)}{q(x)})_{x \sim q(x)} \\
 &\approx \frac{1}{N} \sum_{n=1}^N f(x) \frac{p(x)}{q(x)}_{x \sim q(x)}
 \end{aligned}$$

$$\begin{aligned}
 &\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta}^{GAE}(s_n^t, a_n^t) \nabla \log P_{\theta}(a_n^t | s_n^t) \\
 &= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta'}^{GAE}(s_n^t, a_n^t) \frac{P_{\theta}(a_n^t | s_n^t)}{P_{\theta'}(a_n^t | s_n^t)} \nabla \log P_{\theta}(a_n^t | s_n^t) \\
 &= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta'}^{GAE}(s_n^t, a_n^t) \frac{P_{\theta}(a_n^t | s_n^t)}{P_{\theta'}(a_n^t | s_n^t)} \frac{\nabla P_{\theta}(a_n^t | s_n^t)}{P_{\theta}(a_n^t | s_n^t)} \\
 &= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta'}^{GAE}(s_n^t, a_n^t) \frac{\nabla P_{\theta}(a_n^t | s_n^t)}{P_{\theta'}(a_n^t | s_n^t)} \\
 \text{Loss} &= -\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta'}^{GAE}(s_n^t, a_n^t) \frac{P_{\theta}(a_n^t | s_n^t)}{P_{\theta'}(a_n^t | s_n^t)}
 \end{aligned}$$



这个学生不能和你差距太大。
不然你很难学到对你有用的经验和教训。

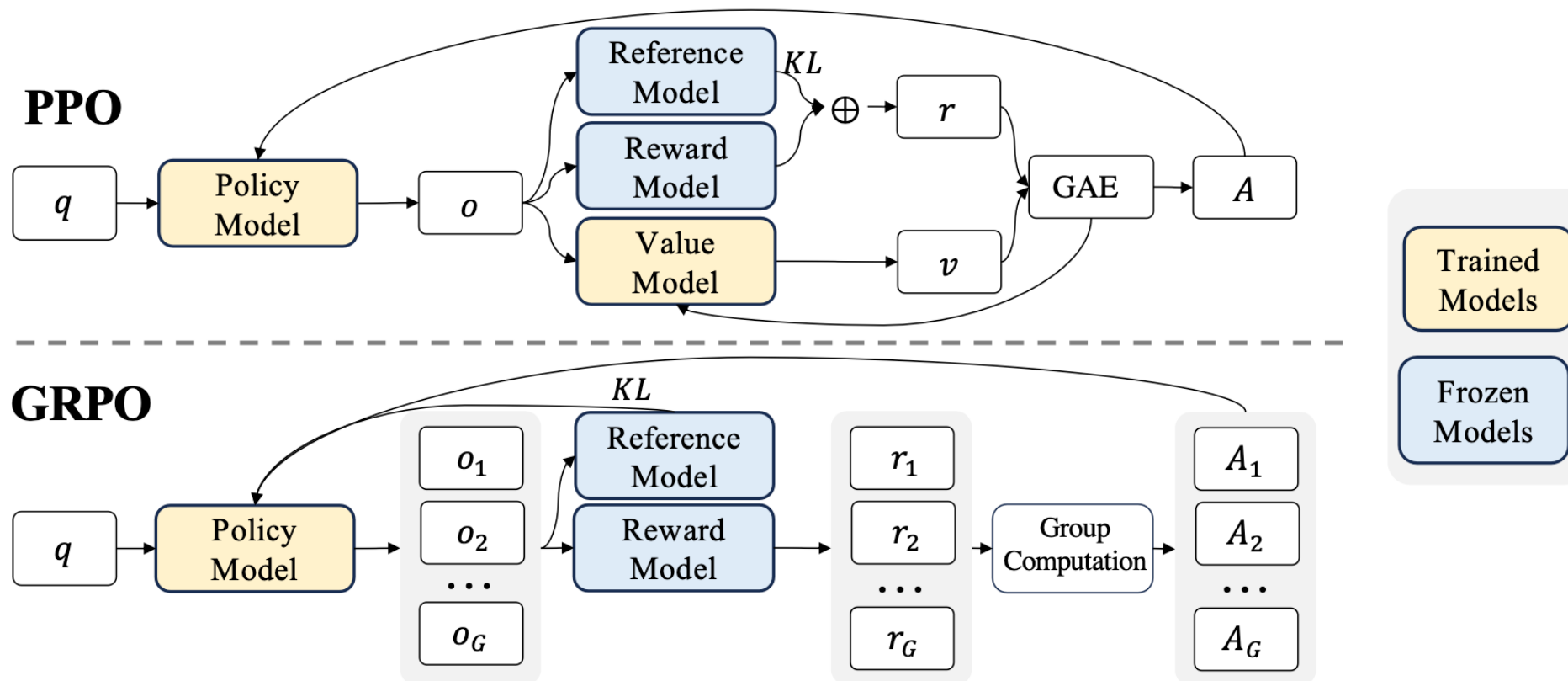
$$Loss_{ppo} = -\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} A_{\theta'}^{GAE}(s_n^t, a_n^t) \frac{P_{\theta}(a_n^t | s_n^t)}{P_{\theta'}(a_n^t | s_n^t)} + \beta KL(P_{\theta}, P_{\theta'})$$

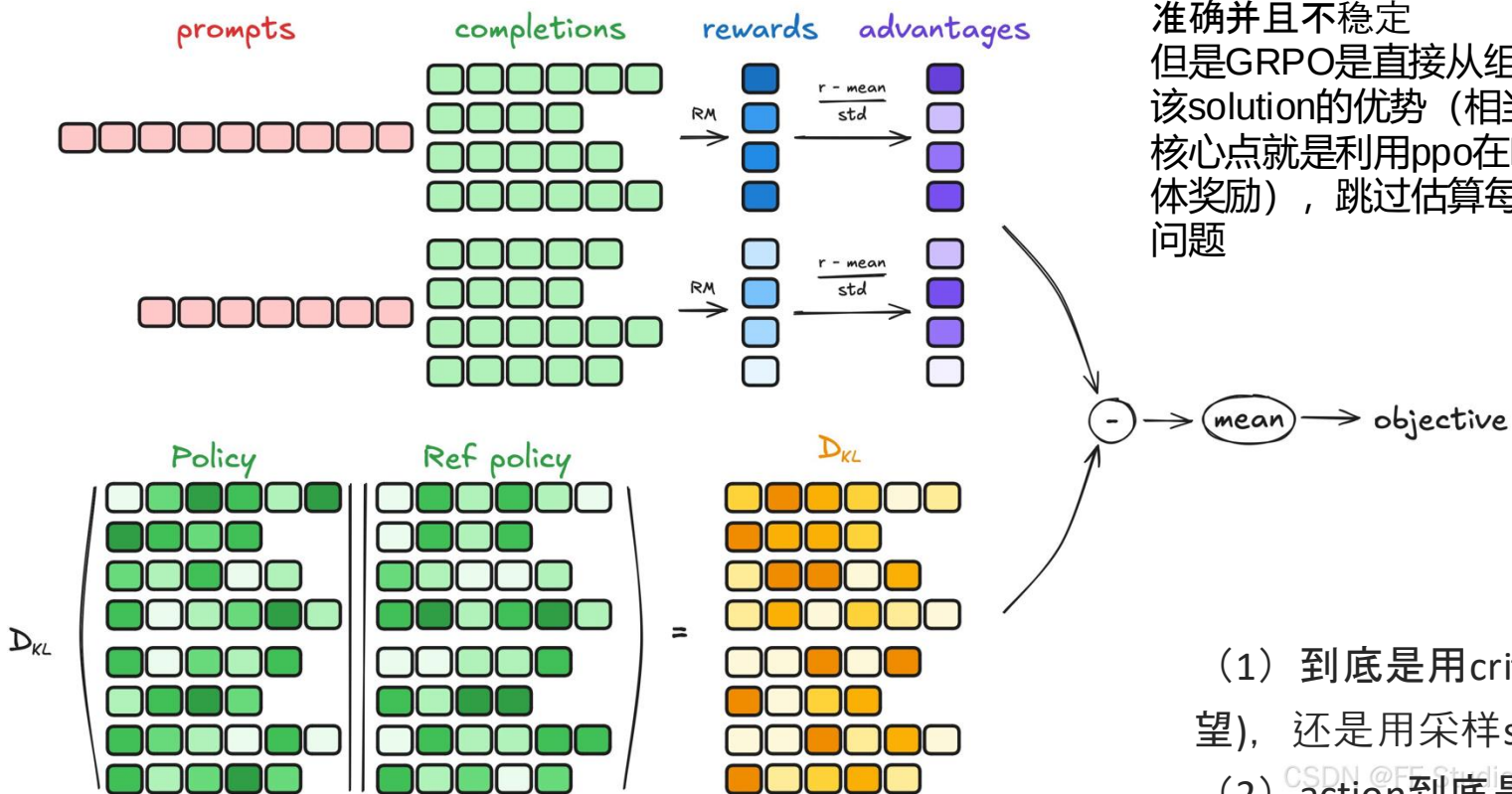
$$Loss_{ppo2} = -\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} \min(A_{\theta'}^{GAE}(s_n^t, a_n^t) \frac{P_{\theta}(a_n^t | s_n^t)}{P_{\theta'}(a_n^t | s_n^t)}, \text{clip}(\frac{P_{\theta}(a_n^t | s_n^t)}{P_{\theta'}(a_n^t | s_n^t)}, 1 - \epsilon, 1 + \epsilon) A_{\theta'}^{GAE}(s_n^t, a_n^t))$$

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

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

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





PPO需要价值函数估计生成每个token的value, 这样可能导致中间估计的不准确并且不稳定

但是GRPO是直接从组内的相对评分来计算优势, 给奖励的时候也是直接给该solution的优势(相当于隐式奖励该solution的所有token)。GRPO的一个核心点就是利用ppo在llm的奖励稀疏性(只给最后一个token/只给solution整体奖励), 跳过估算每一个token的value的过程, 直接从solution-level解决问题

- (1) 到底是用critic model拟合出base (期望), 还是用采样simulate出base。
- (2) action到底是token-level 还是 solution-level的。

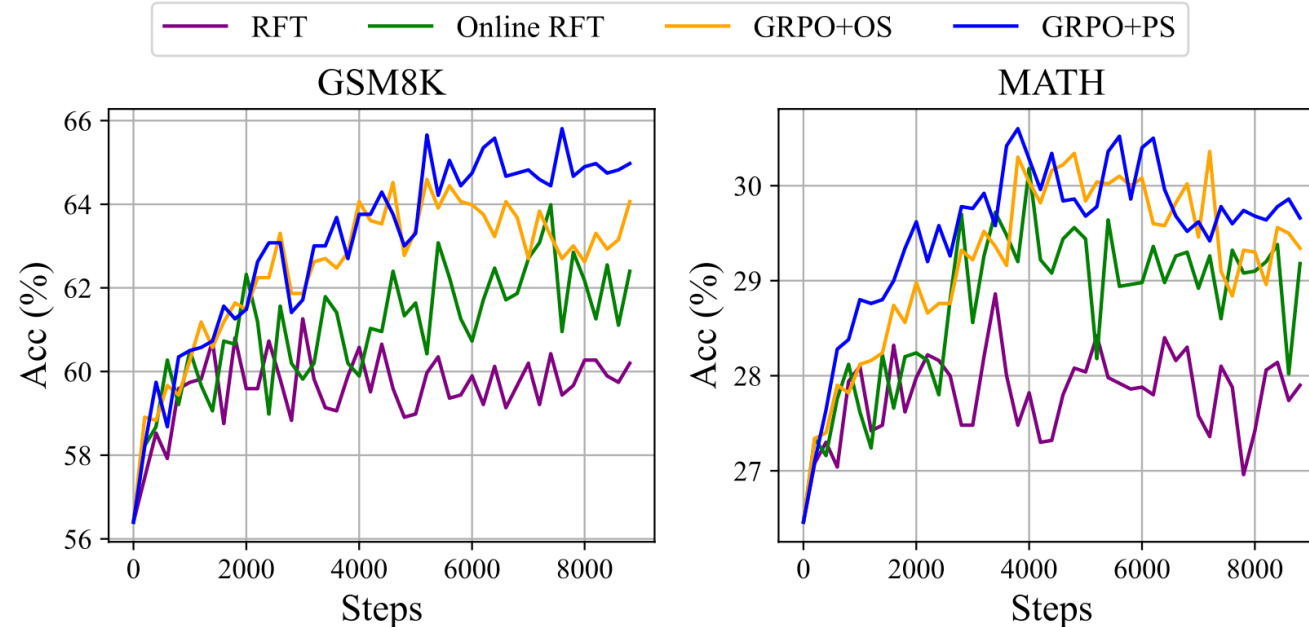
Training Setting	Training Tokens			w/o Tool Use			w/ Tool Use	
	General	Code	Math	GSM8K	MATH	CMATH	GSM8K+Python	MATH+Python
No Continual Training	–	–	–	2.9%	3.0%	12.3%	2.7%	2.3%
Two-Stage Training								
Stage 1: General Training	400B	–	–	2.9%	3.2%	14.8%	3.3%	2.3%
Stage 2: Math Training	–	–	150B	19.1%	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%

Table 6 | Investigation of how code affects mathematical reasoning under different training settings. We experiment with DeepSeek-LLM 1.3B, and evaluate its mathematical reasoning performance without and with tool use via few-shot chain-of-thought prompting and few-shot program-of-thought prompting, respectively.

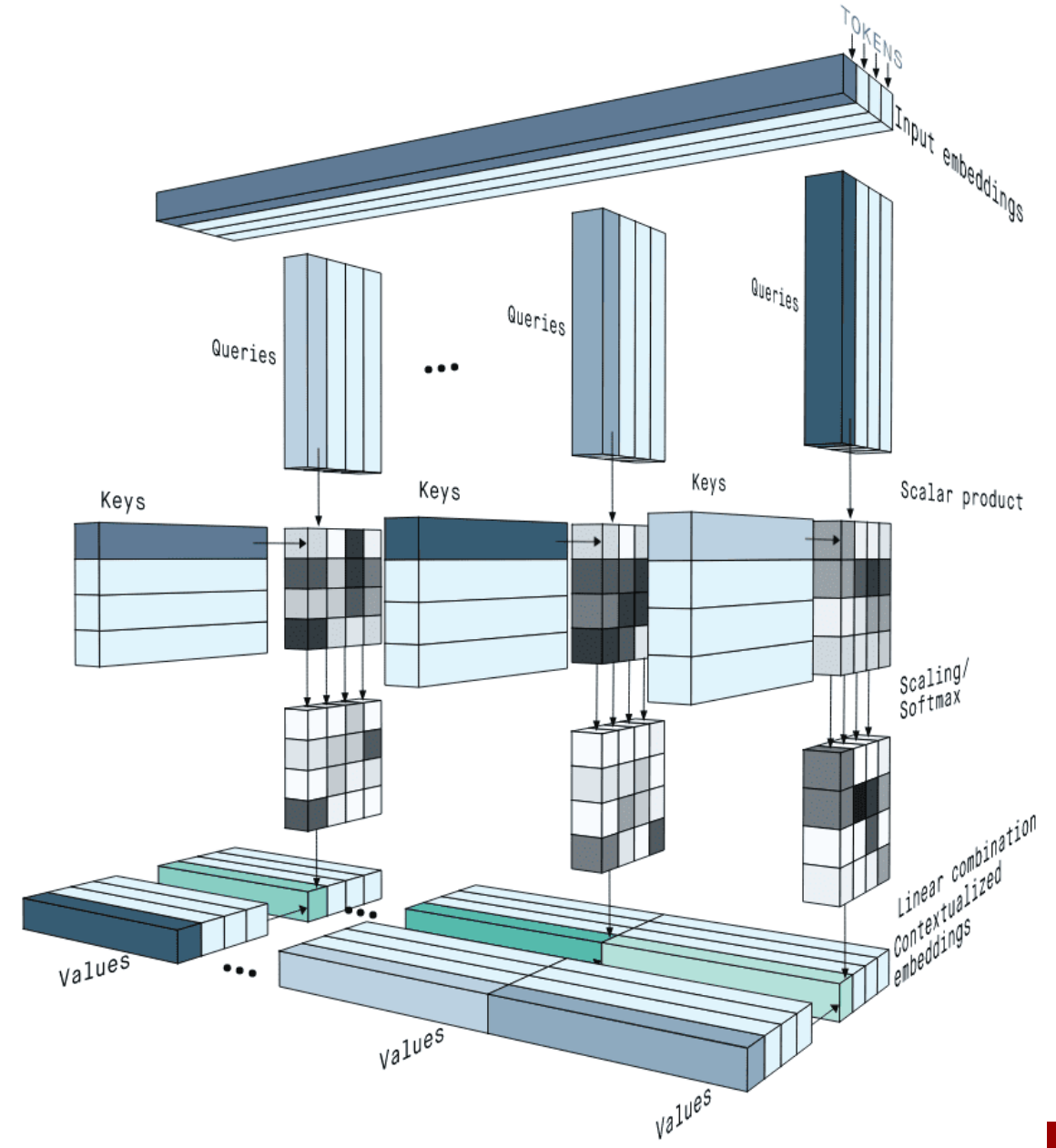
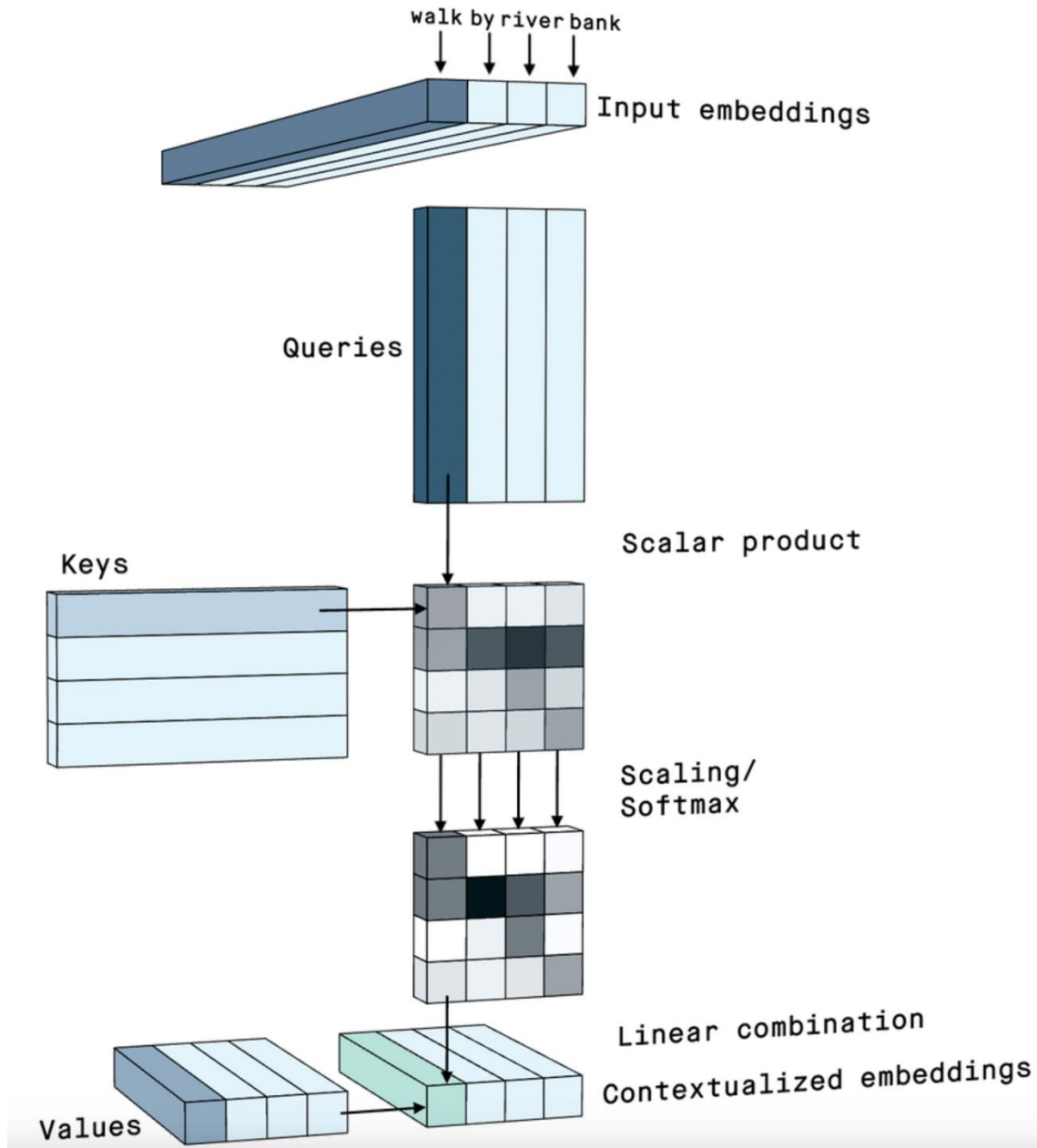
代码训练有助于程序辅助的数学推理，在两阶段训练和单阶段训练设置下都有所体现，码词元与数学词元混合用于单阶段训练时，仍然能够提高数学推理能力，并减轻灾难性遗忘问题

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

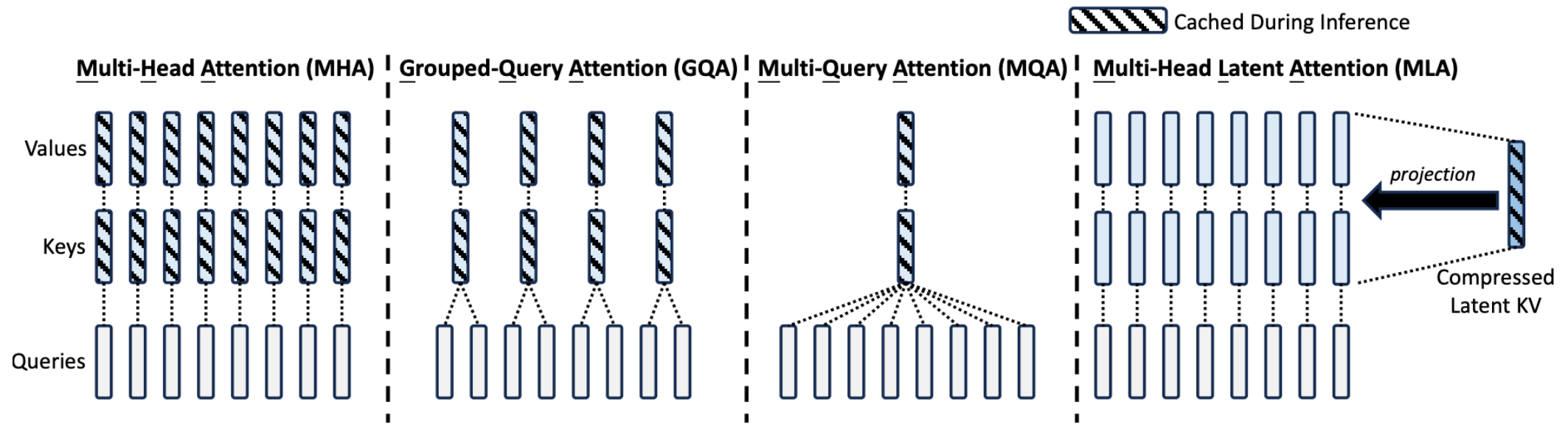
Methods	Data Source	Reward Function	Gradient Coefficient
SFT	$q, o \sim P_{sft}(Q, O)$	-	1
RFT	$q \sim P_{sft}(Q), o \sim \pi_{sft}(O q)$	Rule	Equation 10
DPO	$q \sim P_{sft}(Q), o^+, o^- \sim \pi_{sft}(O q)$	Rule	Equation 14
Online RFT	$q \sim P_{sft}(Q), o \sim \pi_{\theta}(O q)$	Rule	Equation 10
PPO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



From MHA, GQA to MLA



Multi-Head Latent Attention



多头注意力
MHA等价于多个独立的单头注意力的拼接

思想也很朴素，它就是将所有Head分为 g 个组，每组共享同一对K、V

直接让所有Attention Head共享同一个K、V

低秩投影 / 压缩
朴素的思想是只保存一个向量，在推理的时候还原

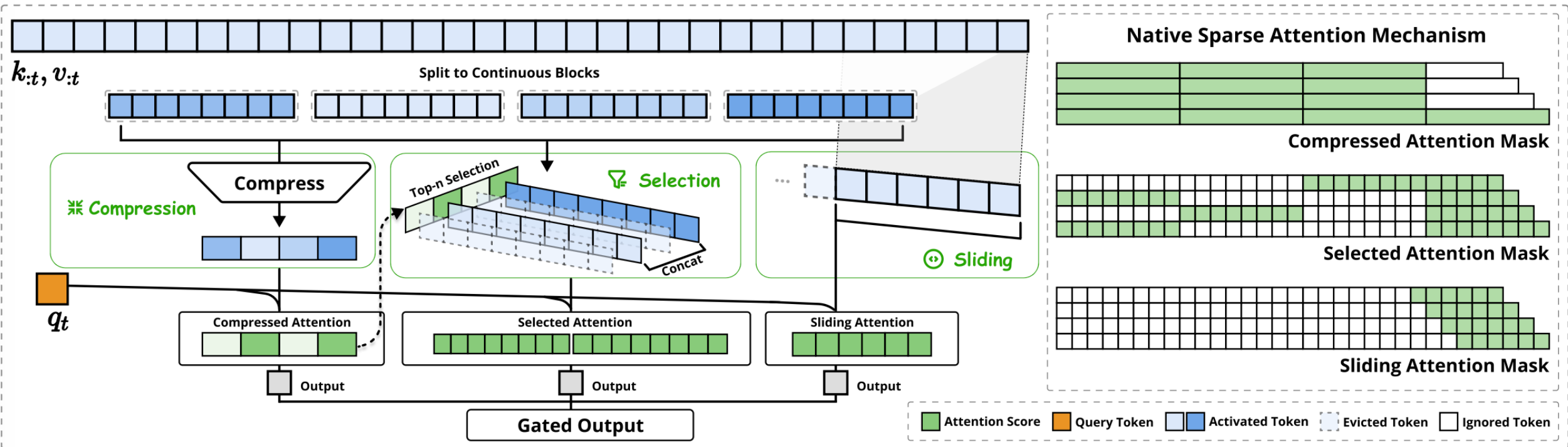
Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention

Jingyang Yuan^{*1,2}, Huazuo Gao¹, Damai Dai¹, Junyu Luo², Liang Zhao¹, Zhengyan Zhang¹, Zhenda Xie¹,
Y. X. Wei¹, Lean Wang¹, Zhiping Xiao³, Yuqing Wang¹, Chong Ruan¹, Ming Zhang², Wenfeng Liang¹,
Wangding Zeng¹

¹DeepSeek-AI

²Key Laboratory for Multimedia Information Processing, Peking University, PKU-Anker LLM Lab

³University of Washington



MTP

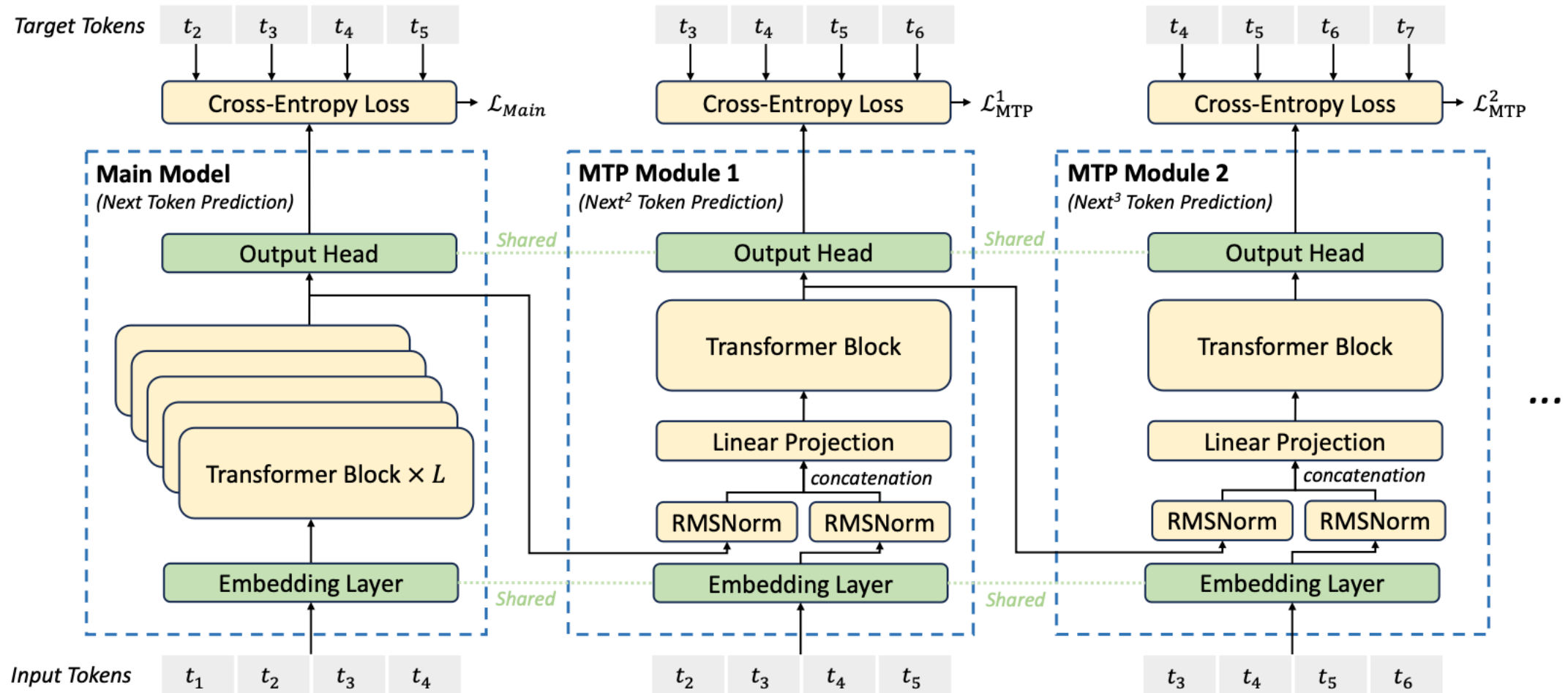


Figure 3 | Illustration of our Multi-Token Prediction (MTP) implementation. We keep the complete causal chain for the prediction of each token at each depth.

DeepSeek LLM Scaling Open-Source Language Models with Longtermism

20240105



Dumps Used	1	2	6	12	16	22	41	91
Deduplication Rate (%)	22.2	46.7	55.7	69.9	75.7	76.3	81.6	89.8

数据上：多个数据集去重

Table 1 | Deduplication ratios for various Common Crawl dumps.

特性	LLaMA	DeepSeek LLM
归一化方法	Pre-Norm + RMSNorm	Pre-Norm + RMSNorm
FFN的激活函数	GELU	SwiGLU
位置编码	RoPE	RoPE
学习率调度器	余弦学习率调度器	多步学习率调度器
注意力机制	MHA (Multi-Head Attention)	GQA (Grouped-Query Attention)
模型扩展策略	扩大FFN层的中间宽度	增加网络深度
训练数据规模	1.4T	2T



计算预算 C 增加时, 模型规模 M 和数据规模 D 如何相应scale

$$M_{\text{opt}} = M_{\text{base}} \cdot C^a, \quad M_{\text{base}} = 0.1715, \quad a = 0.5243$$

$$D_{\text{opt}} = D_{\text{base}} \cdot C^b, \quad D_{\text{base}} = 5.8316, \quad b = 0.4757$$

C 是计算量经验得到的最优batchsize和学习率

$$\eta_{\text{opt}} = 0.3118 \cdot C^{-0.1250}$$

$$B_{\text{opt}} = 0.2920 \cdot C^{0.3271}$$

随着数据质量的提升, 模型扩展指数 a 逐渐增大, 而数据扩展指数 b 逐渐减小, 这表明随着计算预算的增加, 应该更多地分配到模型扩展上, 而不是数据扩展

Approach	Coeff. a where $N_{\text{opt}}(M_{\text{opt}}) \propto C^a$	Coeff. b where $D_{\text{opt}} \propto C^b$
OpenAI (OpenWebText2)	0.73	0.27
Chinchilla (MassiveText)	0.49	0.51
Ours (Early Data)	0.450	0.550
Ours (Current Data)	0.524	0.476
Ours (OpenWebText2)	0.578	0.422

对齐数据的数量和分布

We collect around 1.5 million instruction data instances in English and Chinese, covering a wide range of helpfulness and harmlessness topics. Our helpful data contains 1.2 million instances, with a distribution of 31.2% for general language tasks, 46.6% for mathematical problems, and 22.2% for coding exercises. The safety data consists of 300K instances covering various sensitive topics.

对齐分两个阶段：SFT和DPO;

只使用SFT会出现的问题:

- **过拟合**: 在对67B模型进行4轮微调时, 观察到模型很快达到了性能的上限, 说明其出现了过拟合。这意味着模型在训练集上表现良好, 但在未见过的数据上表现不佳。
- **生成重复响应**: 在微调过程中, 我们评估了生成模型的**重复率**, 随着数学微调数据的增加, 生成的响应开始出现更高的重复率。这是因为数学微调数据可能包含相似的推理模式, 而较弱的模型在面对这些模式时无法有效理解, 从而生成了重复的响应。

使用DPO的原因:

- **减轻重复问题**: 为了应对重复响应问题, 使用了**两阶段微调**和**DPO** (直接偏好优化)。DPO通过优化模型的偏好 (例如, 生成更合适的响应而非重复响应) 来增强模型的生成能力。DPO能够在保持基准分数的同时, 显著减少生成的重复响应。
- **提高模型的开放式生成能力**: DPO有助于改善模型的生成技能, 特别是在生成更具多样性和逻辑性的响应方面。它能够增强模型在推理任务中的表现, 并防止过度依赖已知模式, 避免生成无意义或重复的文本。



小模型需要更长时间在数学和代码数据集上进行微调，但这会影响模型的对话能力，比如增加重复行为。实现了阶段性微调过程。在这种方法中，第一阶段使用所有可用数据进行微调，而第二阶段专门聚焦于对话数据的微调

Model	HumanEval	GSM8K	Repetition	IFEval
DeepSeek LLM 7B Chat Stage1	48.2	63.9	0.020	38.0
DeepSeek LLM 7B Chat Stage2	48.2	63.0	0.014	41.2

Table 12 | Two-stage fine-tuning results. The repetition ratio is computed when the temperature is 0. The lower repetition ratio is better. The IFEval result is the prompt-level loose accuracy.

添加2000万条中文多选题数据， MC）数据被证明不仅对中文多选题基准有益， 还能改善英文基准。这表明模型解决多选题问题的能力得到了增强。

Model	MMLU	C-Eval	CMMLU	TriviaQA	ChineseQA
DeepSeek LLM 7B Chat	49.4	47.0	49.7	57.9	75.0
DeepSeek LLM 7B Chat + MC	60.9	71.3	73.8	57.9	74.4

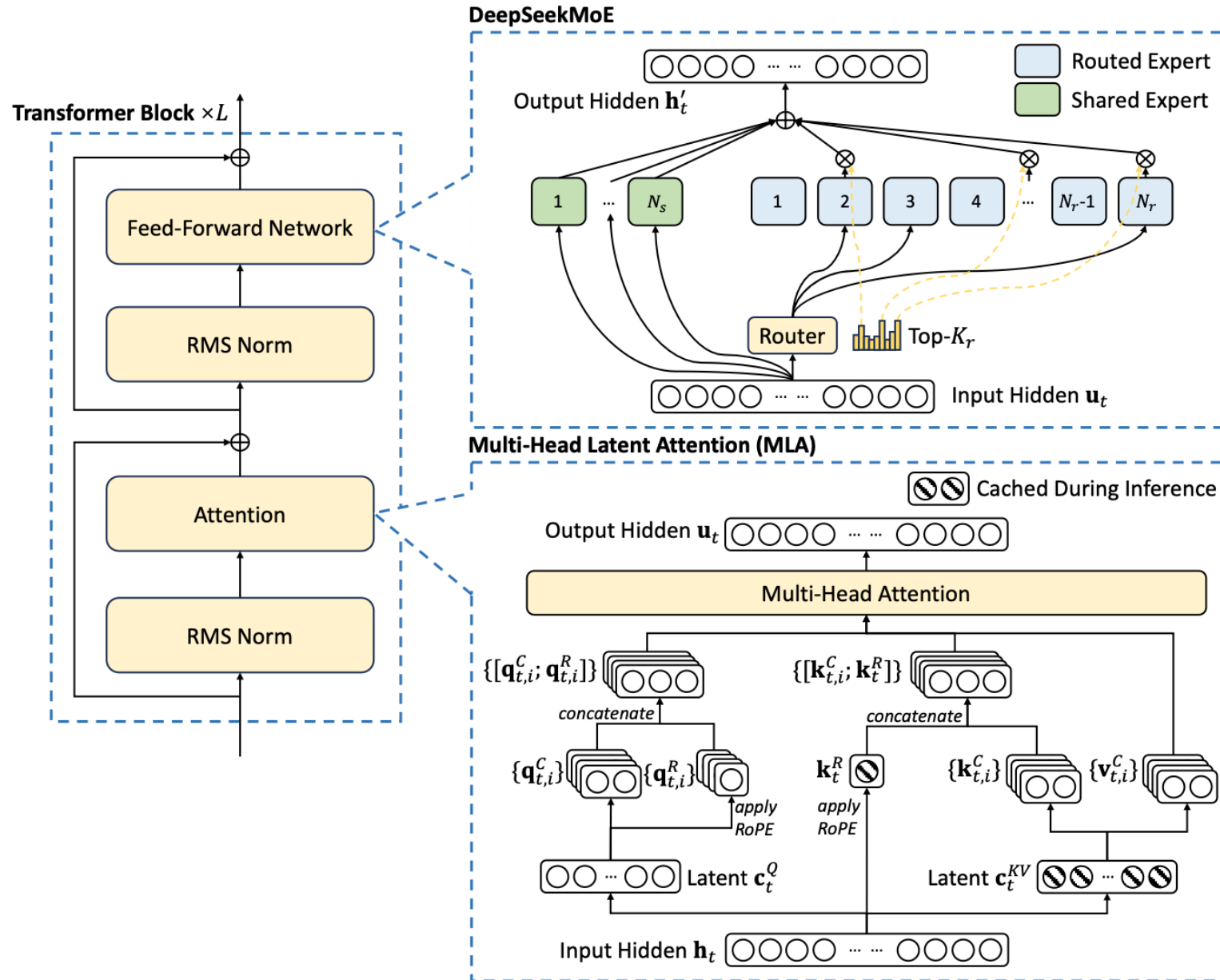
Table 13 | The impact of adding multi-choice question data.

大型模型能够更好地理解系统提示，从而有效地生成更优质的响应。另一方面，小型模型很难充分理解系统提示，训练和测试之间的不一致可能会对其性能产生负面影响。

Model	MT Bench
DeepSeek LLM 7B Chat	7.15
DeepSeek LLM 7B Chat + System Prompt	7.11
DeepSeek LLM 67B Chat	8.35
DeepSeek LLM 67B Chat + System Prompt	8.58

Table 14 | The impact of adding a system prompt.

DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model 20240619



DeepSeek-V3 Technical Report

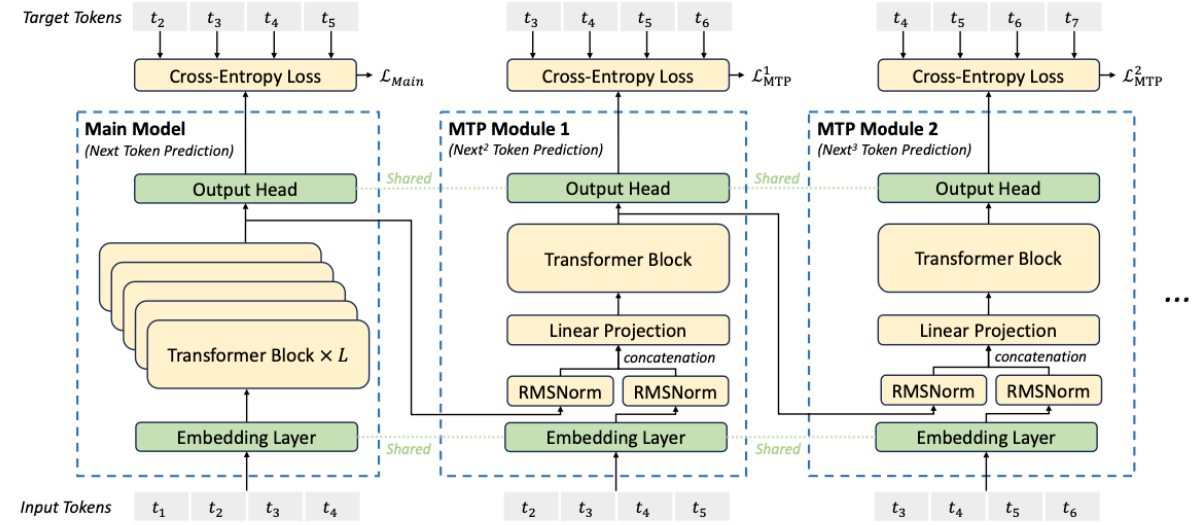
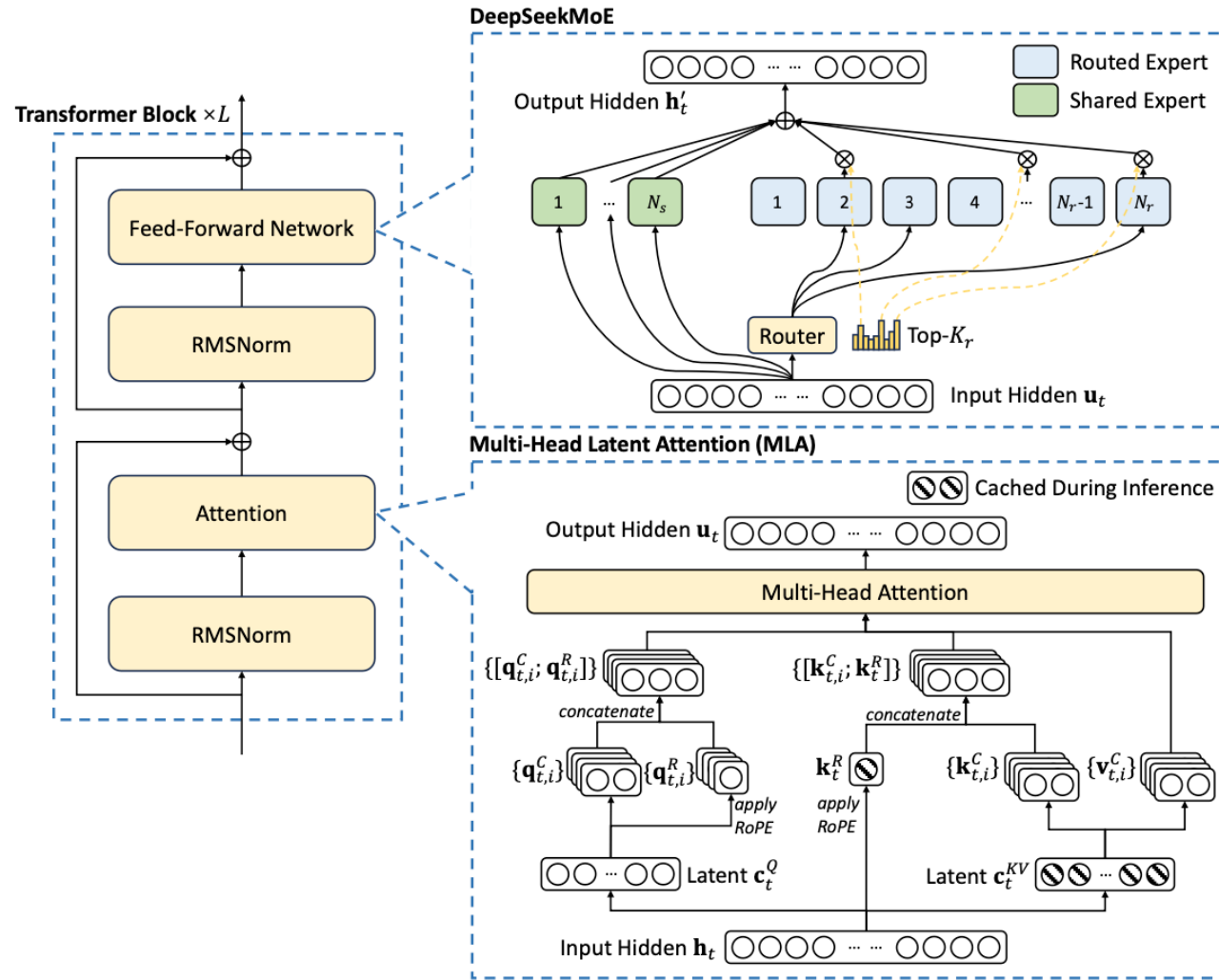


Figure 3 | Illustration of our Multi-Token Prediction (MTP) implementation. We keep the complete causal chain for the prediction of each token at each depth.



14.8T

In the training process of DeepSeekCoder-V2 (DeepSeek-AI, 2024a), we observe that the Fill-in-Middle (FIM) strategy does not compromise the next-token prediction capability while enabling the model to accurately predict middle text based on contextual cues. In alignment with DeepSeekCoder-V2, we also incorporate the FIM strategy in the pre-training of DeepSeek-V3. To be specific, we employ the Prefix-Suffix-Middle (PSM) framework to structure data as follows:

$\langle |f_{im_begin}| \rangle f_{pre} \langle |f_{im_hole}| \rangle f_{suf} \langle |f_{im_end}| \rangle f_{middle} \langle |eos_token| \rangle$.

5.1 监督微调

我们构建的指令微调数据集包含150万条跨领域实例，每个领域采用针对其需求定制的数据生成方法：

推理数据

针对数学、代码竞赛与逻辑谜题等推理相关数据，我们通过内部DeepSeek-R1模型生成数据。尽管R1生成的数据准确性较高，但其存在过度思考、格式混乱与冗长等问题。我们的目标是平衡R1生成数据的高准确性与常规格式化数据的清晰简洁性。具体实现步骤如下：

1. 专家模型构建：

针对代码、数学与通用推理等特定领域，通过监督微调（SFT）与强化学习（RL）联合训练流程构建专家模型，作为最终模型的数据生成器。

2. 双模式SFT样本生成：

- 原始响应模式：以 $\langle \text{问题}, \text{原始响应} \rangle$ 格式生成样本。
- 系统提示模式：以 $\langle \text{系统提示}, \text{问题}, \text{R1响应} \rangle$ 格式生成样本，其中系统提示包含引导模型进行反思与验证的指令。

3. RL阶段优化：

在RL阶段采用**高温采样（high-temperature sampling）**生成融合R1生成数据与原始数据模式的响应，经过数百步训练后，中间RL模型学会整合R1模式以策略性提升性能。

4. 拒绝采样筛选：

通过专家模型进行拒绝采样，筛选高质量SFT数据，确保最终数据既保留DeepSeek-R1的优势，又生成简洁有效的响应。

非推理数据

对于创意写作、角色扮演与简单问答等非推理数据，我们使用DeepSeek-V2.5生成响应，并通过人工标注验证数据的准确性与正确性。

微调参数设置

- 训练轮次：对DeepSeek-V3-Base进行2轮微调。
- 学习率调度：采用余弦衰减策略，初始学习率 5×10^{-6} ，逐步降至 1×10^{-6} 。
- 数据打包：单序列由多个样本打包而成，但采用样本掩码策略确保样本间隔离且不可见。



5.2.1 强化学习

奖励模型

我们在强化学习过程中采用基于规则的奖励模型（RM）和基于模型的奖励模型：

基于规则的RM

对于可通过特定规则验证的问题（如数学题确定性结果或LeetCode代码题），我们要求模型以指定格式（如方框标记）提供最终答案，并通过规则验证正确性。例如：

- 数学问题：验证最终答案是否符合数值解
- 代码问题：利用编译器基于测试用例生成反馈

基于模型的RM

对于自由格式真实答案的问题（如创意写作），奖励模型需基于问题和对应答案生成反馈。该模型从DeepSeek-V3 SFT检查点训练，并通过包含**思维链（chain-of-thought）**的偏好数据增强可靠性，以降低特定任务的奖励黑客攻击风险。

Model	Chat	Chat-Hard	Safety	Reasoning	Average
GPT-4o-0513	96.6	70.4	86.7	84.9	84.7
GPT-4o-0806	96.1	76.1	88.1	86.6	86.7
GPT-4o-1120	95.8	71.3	86.2	85.2	84.6
Claude-3.5-sonnet-0620	96.4	74.0	81.6	84.7	84.2
Claude-3.5-sonnet-1022	96.4	79.7	91.1	87.6	88.7
DeepSeek-V3	96.9	79.8	87.0	84.3	87.0
DeepSeek-V3 (maj@6)	96.9	82.6	89.5	89.2	89.6

Table 8 | Performances of GPT-4o, Claude-3.5-sonnet and DeepSeek-V3 on RewardBench.



Model	LiveCodeBench-CoT		MATH-500	
	Pass@1	Length	Pass@1	Length
DeepSeek-V2.5 Baseline	31.1	718	74.6	769
DeepSeek-V2.5 +R1 Distill	37.4	783	83.2	1510

- 1. 蒸馏提升了性能，但也显著增加了平均响应长度
- 2. 从推理模型进行知识蒸馏为训练后优化提供了一个有前景的方向
- 3. 长链思维蒸馏可能对提升需要复杂推理的其他认知任务的模型性能具有重要价值

Table 9 | The contribution of distillation from DeepSeek-R1. The evaluation settings of LiveCodeBench and MATH-500 are the same as in Table 6.

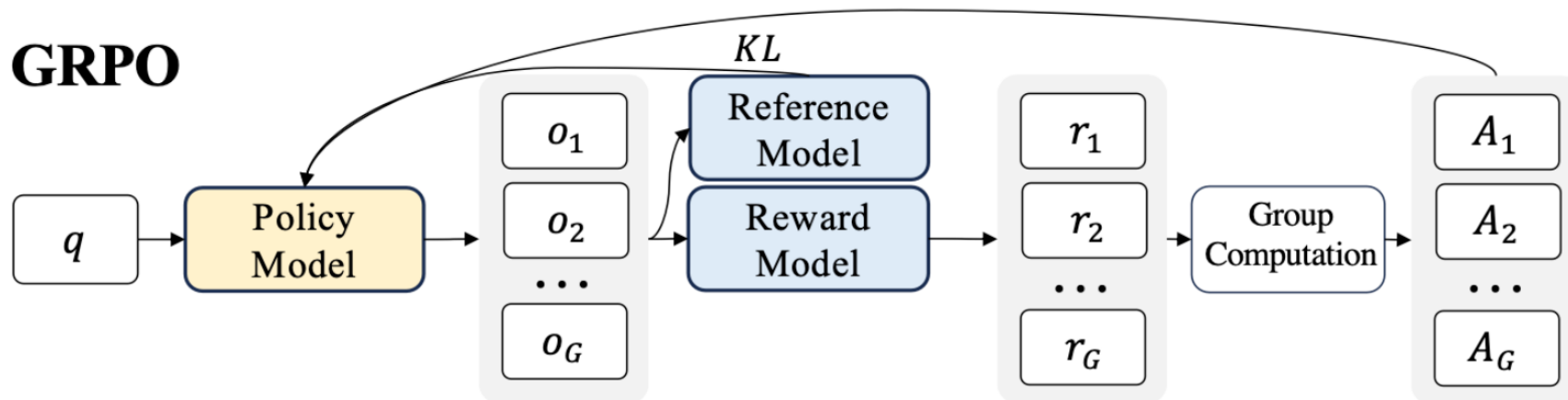
利用DeepSeek-V3自身的投票评估结果作为反馈来源。这种方法产生了显著的对齐效果，显著提升了DeepSeek-V3在主观评估中的表现。 constitutional AI approach

Multi-Token Prediction Evaluation : Second token prediction ranges between 85% and 90%

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

Post-Training: Large-Scale Reinforcement Learning on the Base Model
Distillation: Smaller Models Can Be Powerful Too

GRPO



2.2.2. 奖励建模

奖励是训练信号的来源，决定了强化学习的优化方向。

无奖励模型，纯规则奖励

为了训练DeepSeek-R1-Zero，我们采用了一种基于规则的奖励系统，主要包括两种类型的奖励：

- **准确性奖励**：准确性奖励模型评估响应是否正确。例如，在具有确定性结果的数学问题中，模型需要以指定格式（例如在方框内）提供最终答案，从而能够基于规则可靠地验证正确性。类似地，对于LeetCode问题，可以使用编译器根据预定义的测试用例生成反馈。
- **格式奖励**：除了准确性奖励模型外，我们还采用了格式奖励模型，强制模型将其思考过程放在“<think>”和“</think>”标签之间。

在开发DeepSeek-R1-Zero时，我们没有应用结果或过程的神经奖励模型，因为我们发现神经奖励模型在大规模强化学习过程中可能会受到奖励欺骗的影响，而重新训练奖励模型需要额外的训练资源，并使整个训练流程复杂化。

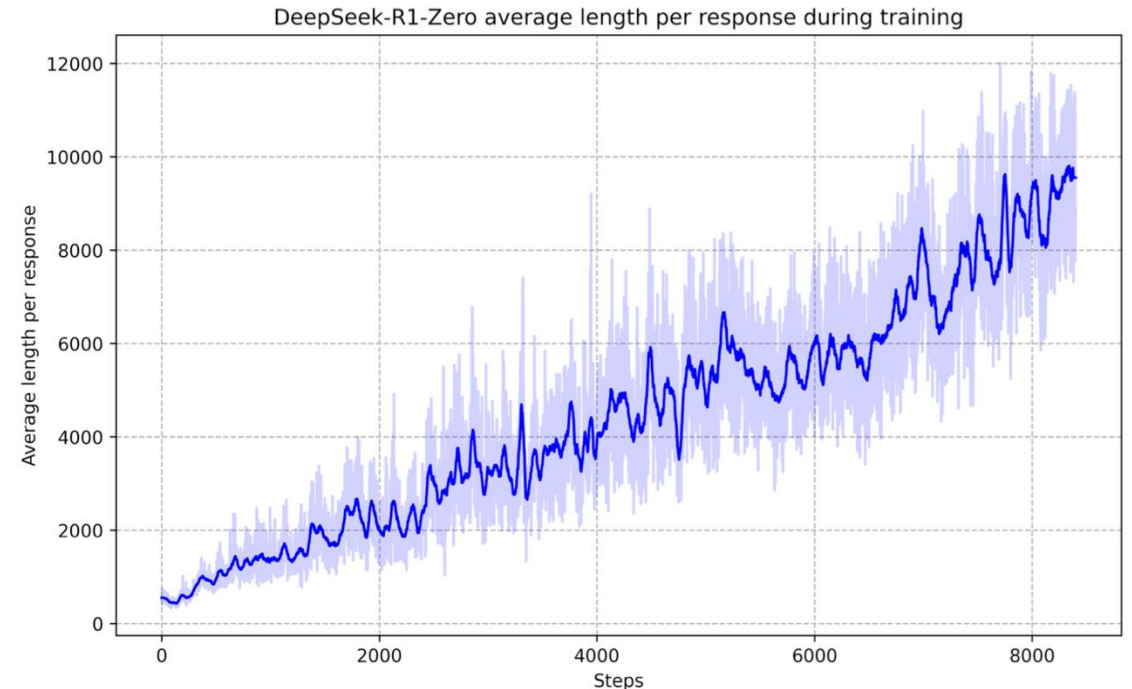
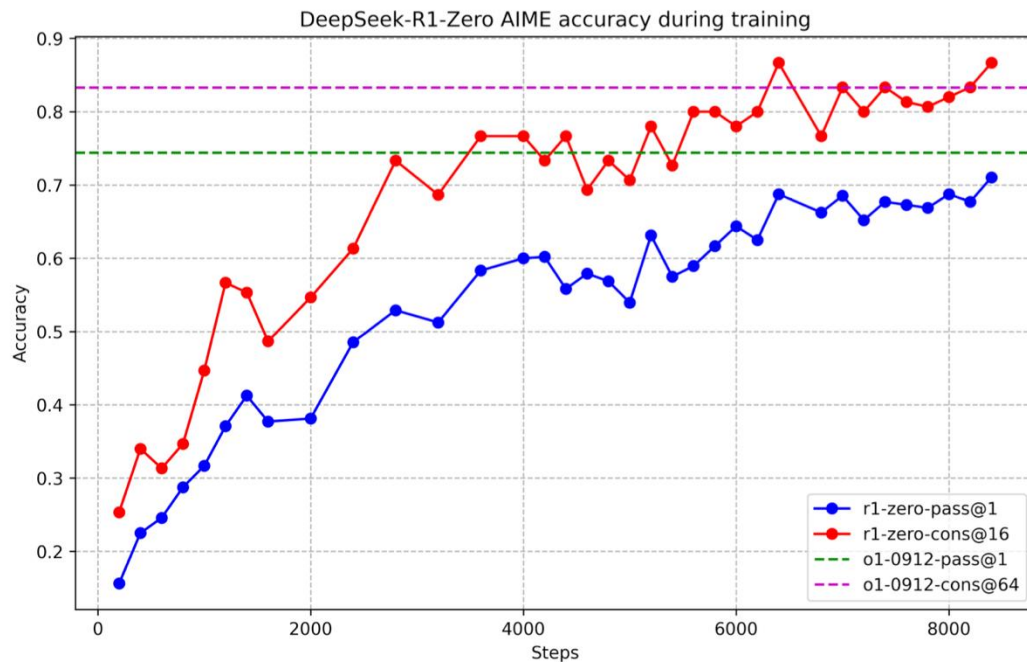
是不是证明奖励模型的路子对不对？

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:

Self-evolution Process of DeepSeek-R1- Zero

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

这种自我进化最显著的一个方面是，随着测试时间计算的增加，复杂行为的涌现。例如，反思行为（模型重新审视和评估其先前步骤）以及探索替代问题解决方法的出现都是自发的。这些行为并非显式编程，而是模型与强化学习环境交互的结果。这种自发发展显著增强了DeepSeek-R1-Zero的推理能力，使其能够更高效、更准确地应对更具挑战性的任务。



Aha Moment of DeepSeek-R1-Zero A particularly intriguing phenomenon observed during the training of DeepSeek-R1-Zero is the occurrence of an “aha moment”. This moment, as illustrated in Table 3, occurs in an intermediate version of the model. During this phase, DeepSeek-R1-Zero learns to allocate more thinking time to a problem by reevaluating its initial approach. This behavior is not only a testament to the model’s growing reasoning abilities but also a captivating example of how reinforcement learning can lead to unexpected and sophisticated outcomes.

This moment is not only an “aha moment” for the model but also for the researchers observing its behavior. It underscores the power and beauty of reinforcement learning: rather than explicitly teaching the model on how to solve a problem we simply provide it with the right incentives and it autonomously develops advanced problem-solving strategies. The “aha moment” serves as a powerful reminder of the potential of RL to unlock new levels of intelligence in artificial systems, paving the way for more autonomous and adaptive models in the future.

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 ...

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

Rearrange to isolate the inner square root term:

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

...

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 ...

We started with the equation:

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

First, let's square both sides:

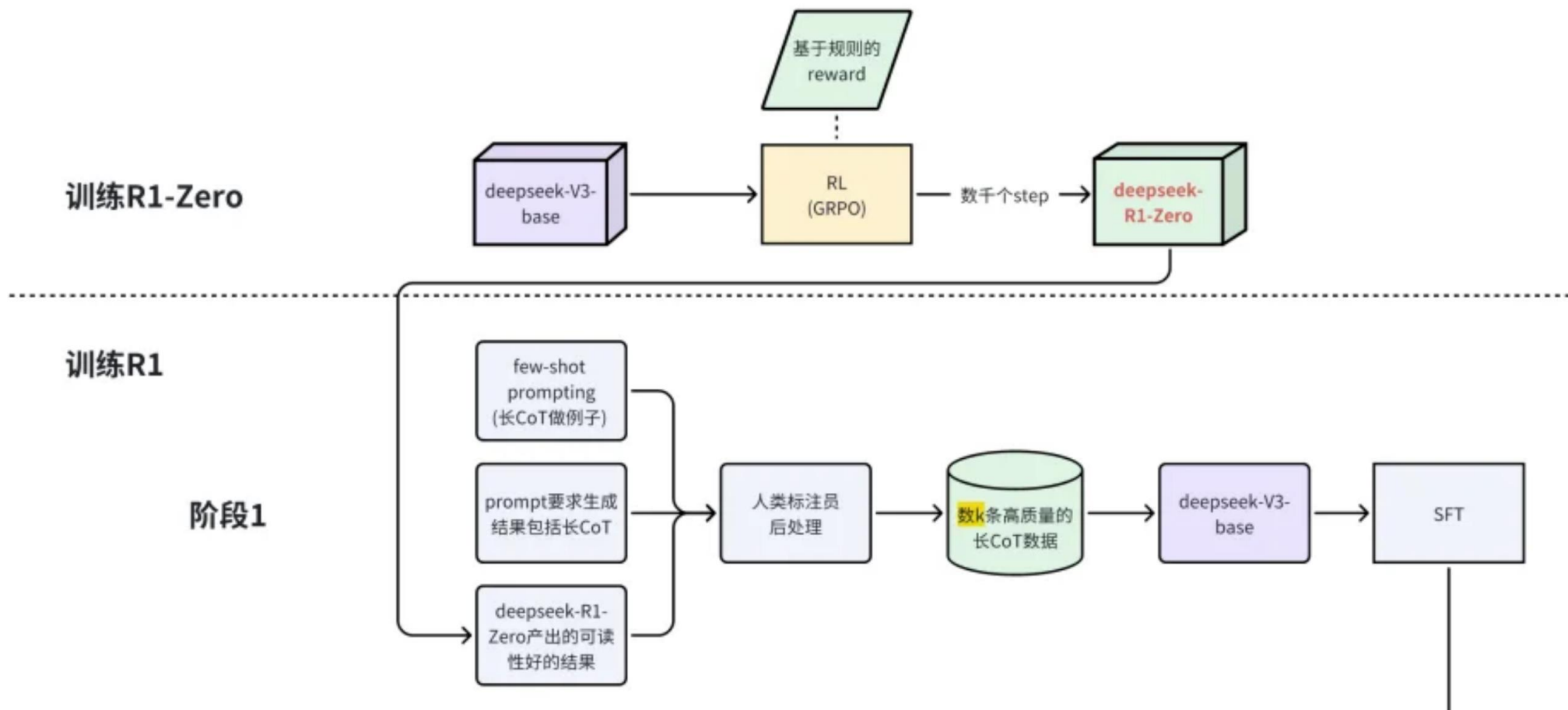
$$a - \sqrt{a + x} = x^2 \implies \sqrt{a + x} = a - x^2$$

Next, I could square both sides again, treating the equation: ...

...

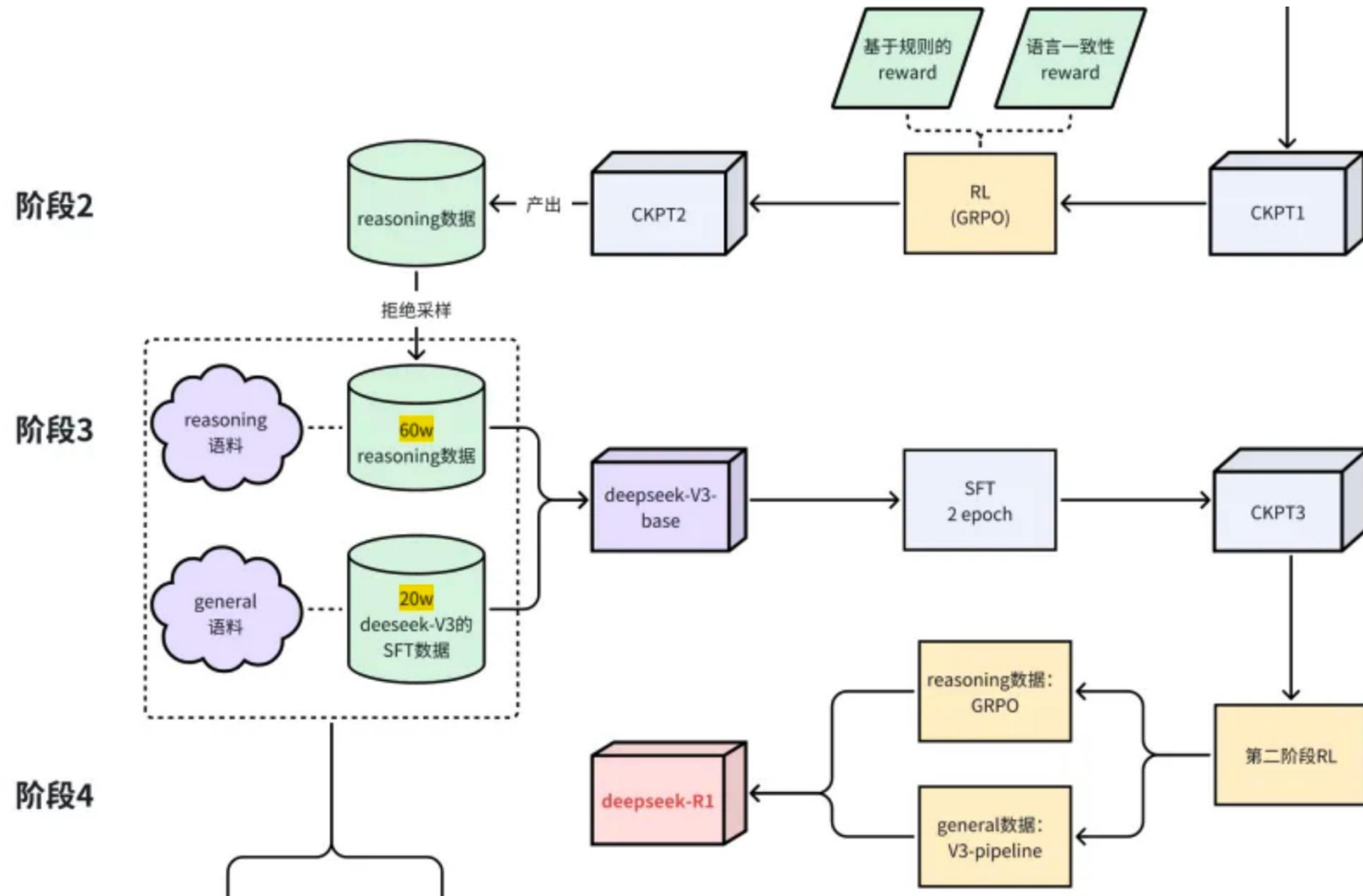
DeepSeek-R1-Zero在可读性差和语言混杂等方面存在挑战

DeepSeek-R1: Reinforcement Learning with Cold Start



- **可读性:** DeepSeek-R1-Zero的一个关键局限是其内容通常不适合阅读。响应可能混杂多种语言或缺乏用于突出答案的Markdown格式。相比之下，在为DeepSeek-R1创建冷启动数据时，我们设计了一种可读的模式，包括在每段响应末尾添加总结，并过滤掉对读者不友好的响应。我们定义输出格式为 `|special_token|<reasoning_process>|special_token|<summary>`，其中推理过程是查询的CoT，总结用于概括推理结果。
- **潜力:** 通过精心设计基于人类先验的冷启动数据模式，我们观察到其性能优于DeepSeek-R1-Zero。我们相信迭代训练是推理模型的更好方式。

DeepSeek-R1: Reinforcement Learning with Cold Start



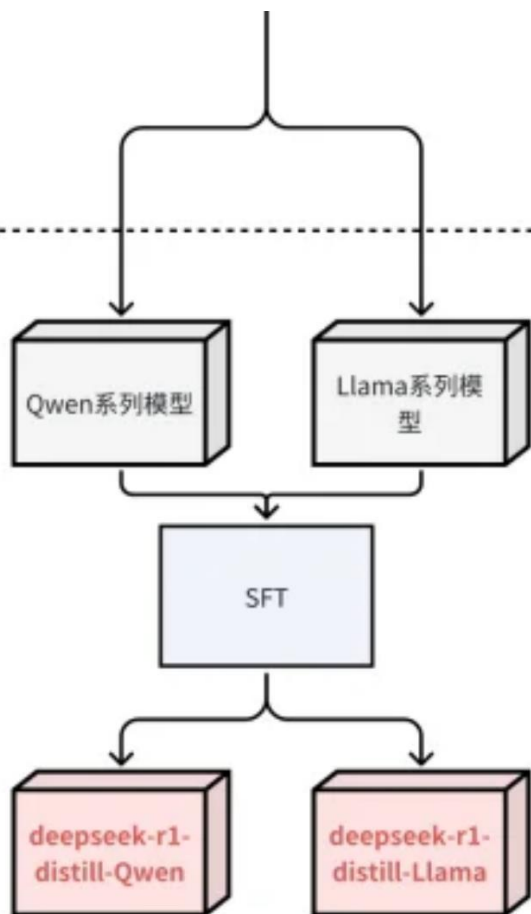
与初始冷启动数据主要关注推理不同，这一阶段引入了其他领域的的数据，以增强模型在写作、角色扮演和其他通用任务中的能力

对于通用数据，我们采用奖励模型捕捉复杂和微妙场景中的人类偏好

蒸馏模型：我们仅应用SFT，未包含RL阶段，
尽管引入RL可以显著提升模型性能。我们的
主要目标是展示蒸馏技术的有效性

阶段4

蒸馏模型



Model	AIME 2024		MATH-500	GPQA Diamond	LiveCode Bench	CodeForces
	pass@1	cons@64				
GPT-4o-0513	9.3	13.4	74.6	49.9	32.9	759
Claude-3.5-Sonnet-1022	16.0	26.7	78.3	65.0	38.9	717
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	1820
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9	1316
DeepSeek-R1-Distill-Qwen-1.5B	28.9	52.7	83.9	33.8	16.9	954
DeepSeek-R1-Distill-Qwen-7B	55.5	83.3	92.8	49.1	37.6	1189
DeepSeek-R1-Distill-Qwen-14B	69.7	80.0	93.9	59.1	53.1	1481
DeepSeek-R1-Distill-Qwen-32B	72.6	83.3	94.3	62.1	57.2	1691
DeepSeek-R1-Distill-Llama-8B	50.4	80.0	89.1	49.0	39.6	1205
DeepSeek-R1-Distill-Llama-70B	70.0	86.7	94.5	65.2	57.5	1633

Table 5 | Comparison of DeepSeek-R1 distilled models and other comparable models on reasoning-related benchmarks.



Discussion

4.1. Distillation v.s. Reinforcement Learning

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCodeBench
	pass@1	cons@64	pass@1	pass@1	pass@1
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9
DeepSeek-R1-Zero-Qwen-32B	47.0	60.0	91.6	55.0	40.2
DeepSeek-R1-Distill-Qwen-32B	72.6	83.3	94.3	62.1	57.2

Table 6 | Comparison of distilled and RL Models on Reasoning-Related Benchmarks.

Unsuccessful Attempts

过程奖励模型（PRM）

PRM是一种合理的方法，可以引导模型采用更好的方法解决推理任务（Lightman et al., 2023; Uesato et al., 2022; Wang et al., 2023）。然而，在实践中，PRM有三个主要局限可能阻碍其最终成功：首先，在通用推理中明确定义细粒度步骤具有挑战性，其次，判断当前中间步骤是否正确是一项困难的任务，使用模型进行自动标注可能无法得到令人满意的结果，而人工标注则不利于规模化，第三，一旦引入基于模型的PRM，不可避免地会导致奖励欺骗（Gao et al., 2022），而重新训练奖励模型需要额外的训练资源，并使整个训练流程复杂化。总之，尽管PRM在重排模型生成的Top-N响应或辅助引导搜索（Snell et al., 2024）方面表现出色，但在我们实验中的大规模强化学习过程中，其优势相比引入的额外计算开销有限。

蒙特卡洛树搜索 (MCTS)

受AlphaGo (Silver et al., 2017b) 和AlphaZero (Silver et al., 2017a) 的启发, 我们探索了使用蒙特卡洛树搜索 (MCTS) 来增强测试时计算的可扩展性。这种方法将答案分解为更小的部分, 使模型能够系统地探索解决方案空间。为此, 我们提示模型生成多个标签, 这些标签对应于搜索所需的特定推理步骤。在训练中, 我们首先使用收集的提示通过MCTS在预训练的价值模型指导下找到答案, 随后使用生成的问题-答案对训练策略模型和价值模型, 并迭代优化这一过程。

然而, 这种方法在扩展训练时面临几个挑战: 首先, 与象棋中相对明确的搜索空间不同, 标记生成呈现指数级更大的搜索空间。为了解决这个问题, 我们为每个节点设置了最大扩展限制, 但这可能导致模型陷入局部最优。其次, 价值模型直接影响生成质量, 因为它指导搜索过程的每一步。训练一个细粒度的价值模型本身就很困难, 这使得模型难以迭代改进。尽管AlphaGo的核心成功依赖于训练价值模型以逐步提升性能, 但由于标记生成的复杂性, 这一原则在我们的设置中难以复制。

□ 通用能力不足：做trade还是CL的范围

- DeepSeek-R1在函数调用、多轮对话、复杂**e-off** / 角色扮演和JSON输出等任务上表现不如DeepSeek-V3
- 多阶段训练（冷启动→RL→SFT→二次RL）

□ 蒸馏模型的潜力未完全开发

- 虽然蒸馏小模型表现不错，但未结合RL进一步优化

□ MoE + rl inference