Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model?

Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model?

Yang Yue[1*†], **Zhiqi Chen**[1*], **Rui Lu**[1], **Andrew Zhao**[1], **Zhaokai Wang**[2], **Yang Yue**[1], **Shiji Song**[1], **and Gao Huang**[1✉]

[1] LeapLab, Tsinghua University [2] Shanghai Jiao Tong University

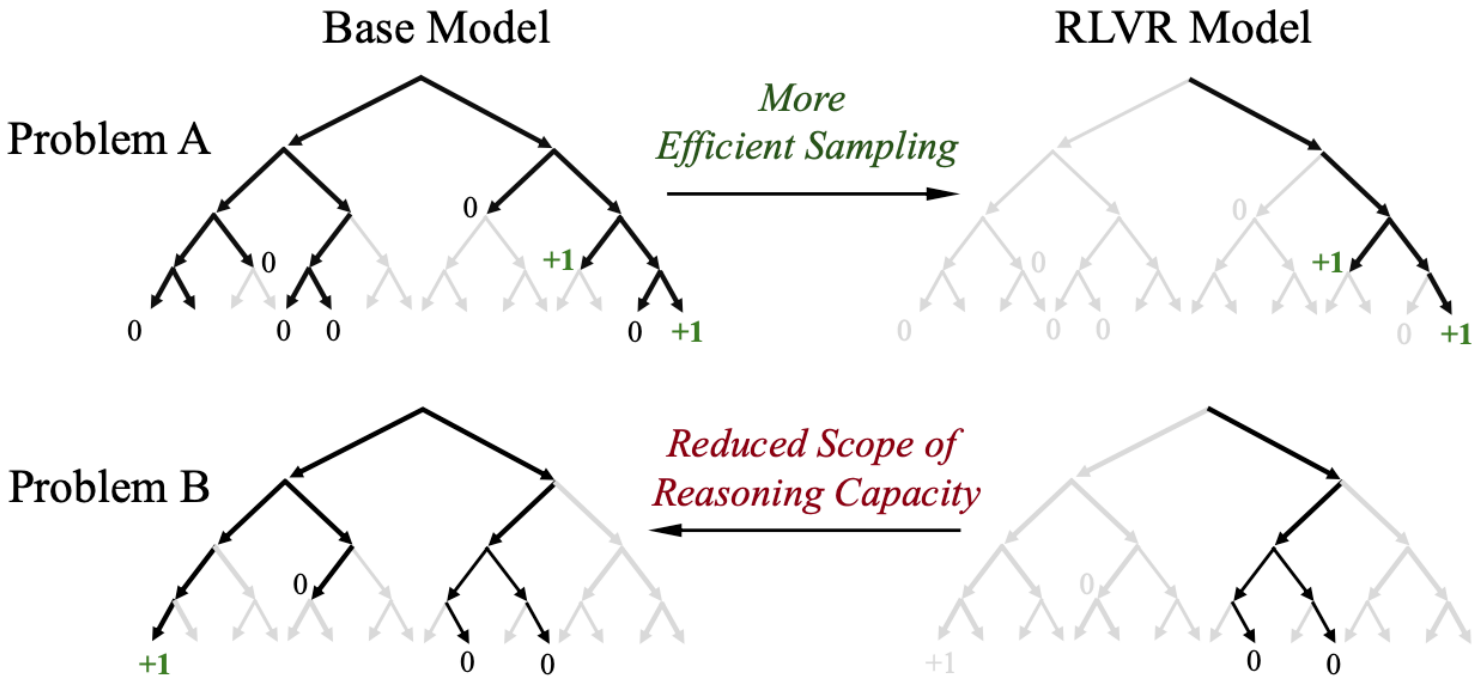
* Equal Contribution † Project Lead ✉ Corresponding Author

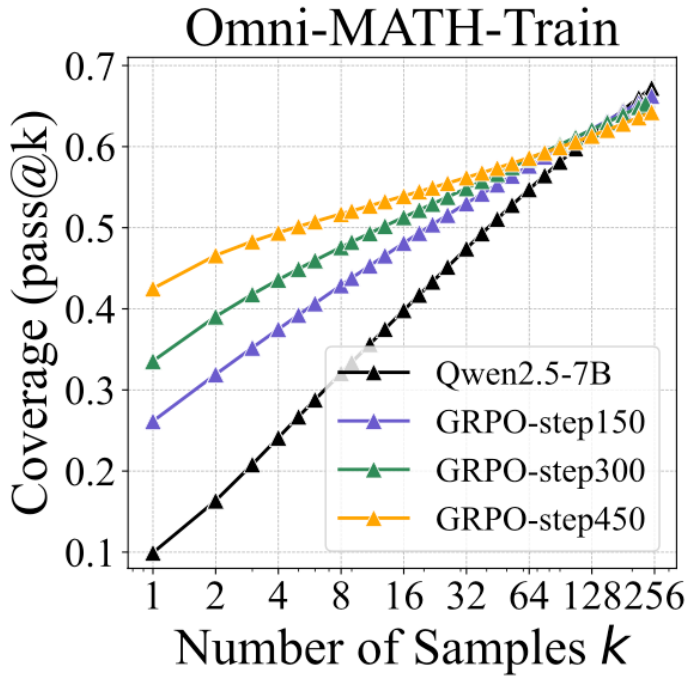
Reinforcement Learning with Verifiable Rewards, RLVR

- **广泛观点认为：RLVR 使得 LLMs 能够不断自我提升，从而习得超越对应基模型的新型推理能力**
- **But 这篇论文的观点 RLVR 训练实际上并未激发出根本性的新型推理模式。**

Question: Does RLVR really bring novel reasoning capabilities to LLMs? If so, what does the model learn from RLVR training?

1. **推理能力**边界：评估方式采用单次成功率或平均采样策略可能低估了模型的真实潜力：若模型在数次尝试后仍未解题，可能并非其无法解决，而是因采样次数不够，所以采用了pass@k
2. **如果允**许基模型大量采样，其性能是否可与 RLVR **模型匹**敌？
3. **涵盖数学、代**码生成和视觉推理，多个任务





Preliminaries

$$\bar{J}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{\mathbf{y}\sim\pi_\theta(\cdot|x)}[\,r\,]\right] \qquad A_i = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}$$

$$\mathcal{L}_{\text{CLIP}} = \mathbb{E}\left[\min(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)\right],$$

Policy Gradient　**最大化生成正确答案**样本的对数似然，同时最小化生成错误答案样本的对数似然。

Zero RL Training　**直接在base模型上**进行强化学习，而不进行任何基于 CoT**的**监督微调

pass@k**的无偏估**计
（实际采样大于k）

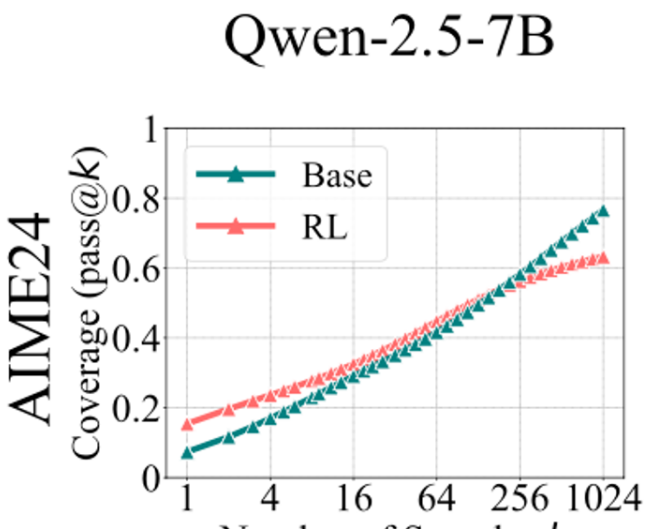
$$\text{pass@}k := \mathbb{E}_{x_i\sim\mathcal{D}}\left[1 - \frac{\binom{n-c_i}{k}}{\binom{n}{k}}\right]$$

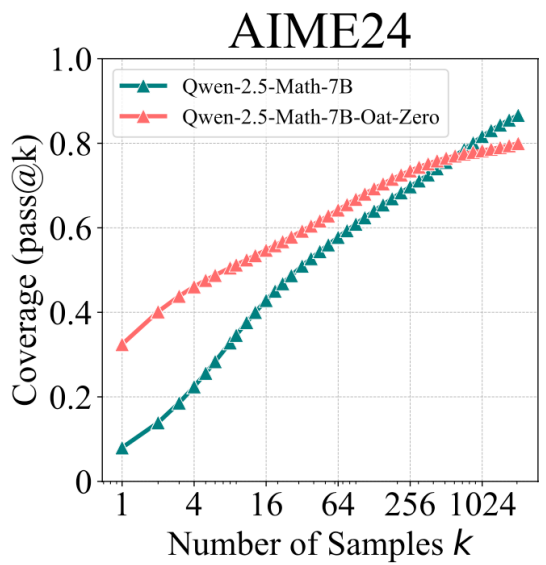
1. 对于coding**任**务，直接使用编译器来验证
2. 对于math**任**务，防止是碰巧猜对的答案，筛除那些容易被**“破解”的**问题，并对部**分模型**输出的推理链（CoT）进行人工检查

Task

Task	Start Model	RL Framework	RL Algorithm(s)	Benchmark(s)
Mathematics	LLaMA-3.1-8B Qwen-2.5-7B/14B/32B-Base Qwen-2.5-Math-7B	SimpleRLZoo Oat-Zero	GRPO	GSM8K, MATH500 Minerva, Olympiad AIME24, AMC23
Code Generation	Qwen-2.5-7B-Instruct	Code-R1	GRPO	LiveCodeBench HumanEval+
Visual Reasoning	Qwen-2.5-VL-7B	EasyR1	GRPO	MathVista MathVision
Deep Analysis	Qwen-2.5-7B-Base Qwen-2.5-7B-Instruct DeepSeek-R1-Distill-Qwen-7B	VeRL	PPO, GRPO Reinforce++ RLOO, ReMax, DAPO	Omni-Math-Rule MATH500

- K比较小的时候，RL > Base：RLVR 能提升采样到正确答案的概率。
- 随着K增大，base的 pass@k 曲线上升斜率明显大于 RL 模型
- 足够大的 k 值下, Base > RL：base模型在可解问题上的覆盖范围更广





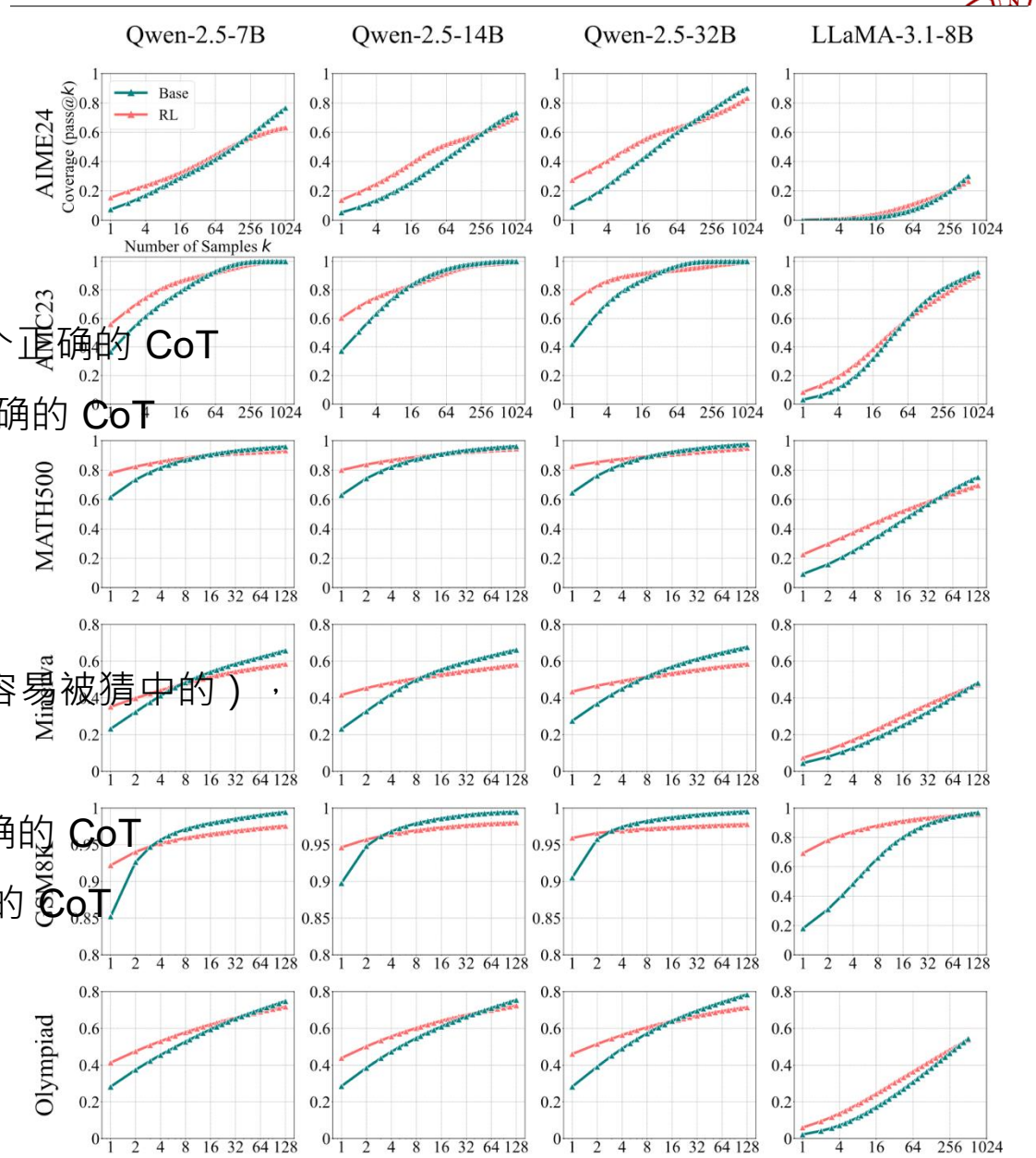
Math

推理链的有效性：

- GSM8K的数据集：挑选准确率<5%的问题，
- base模型共解出 25 个此类问题，其中 24 个问题至少包含一个正确的 CoT
- RL 模型也解出了 25 个问题，其中 23 个问题包含至少一个正确的 CoT
- 解题主要依赖于采样出有效的推理路径，而非偶然猜中

推理链的有效性：

- AIME24的数据集：挑选准确率<5%的问题（过滤：太简单和容易被猜中的），一共18道
- base模型共解出 7个此类问题，其中5个问题至少包含一个正确的 CoT
- RL 模型也解出了 6 个问题，其中 4 个问题包含至少一个正确的 CoT
- 解题主要依赖于采样出有效的推理路径，而非偶然猜中



Code & Visual

由于通过所有单元测试几乎不可能凭猜测完成，pass@k 能够可靠衡量模型的推理边界

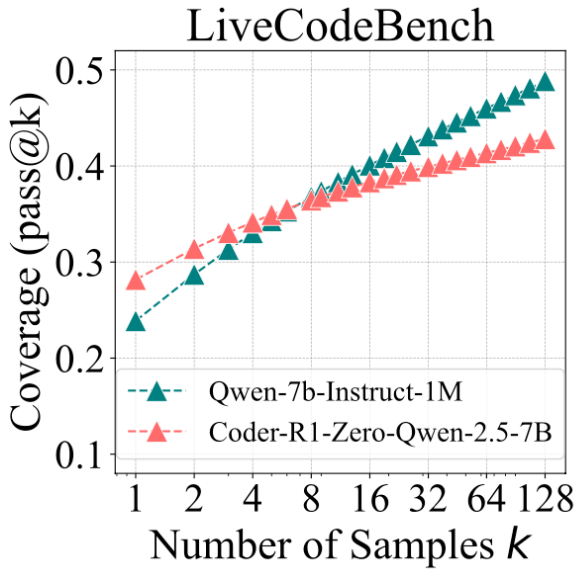


Figure 4: RLVR for Coding.

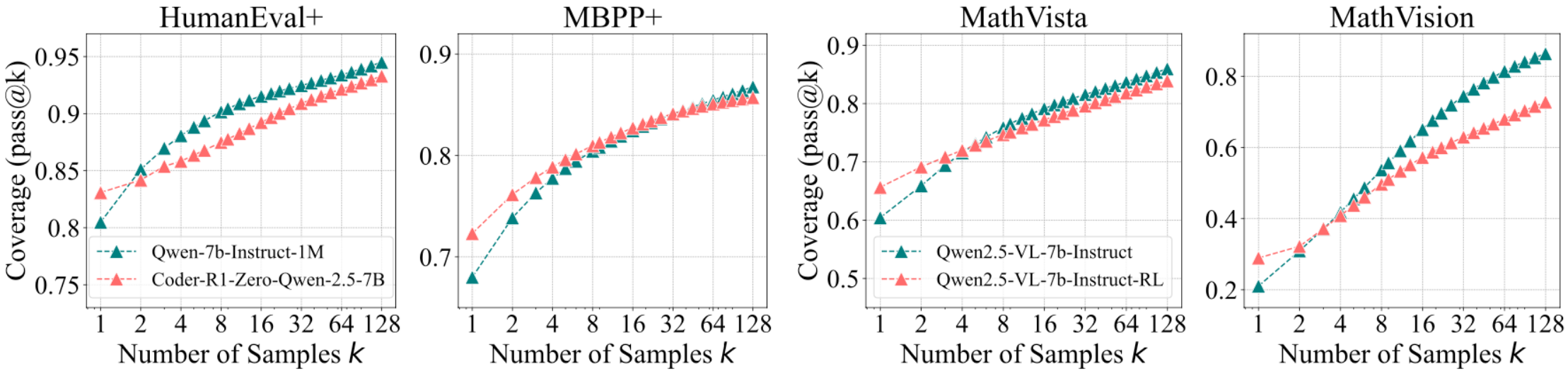


Figure 5: Pass@k curves of base models and zero-RL counterparts. **(Left)** Code Generation. **(Right)** Visual Reasoning.

Deep Analysis

Table 4: Indices of solvable problems in AIME24 (starting from 0). An approximate subset relationship can be observed: most problems solved by the RL model are also solvable by the base model.

Models	Problem Indices
Qwen-7B-Base	0, 1, 4, 6, 7, 8, 9, 11, 12, 14, 15, 16, 17, 18, 19, 22, 23, 24, 25, 26, 27, 28, 29
SimpleRL-Qwen-7B	0, 1, 6, 7, 8, 9, 12, 14, 15, 16, 18, 22, 23, 24, 25, 26, 27, 28, 29

Table 5: Indices of solvable problems in LiveCodeBench (ranging from 400 to 450, starting from 0).

Model	Solvable Problem Indices
Qwen-7B-Instruct-1M	400, 402, 403, 407, 409, 412, 413, 417, 418, 419, 422, 423, 427, 432, 433, 436, 438, 439, 440, 444, 445, 448, 449
Coder-R1	400, 402, 403, 407, 412, 413, 417, 418, 419, 422, 423, 427, 430, 433, 438, 439, 440, 444, 445, 449

RLVR 模型所能解决的问题范围是基模型覆盖范围的近似子集

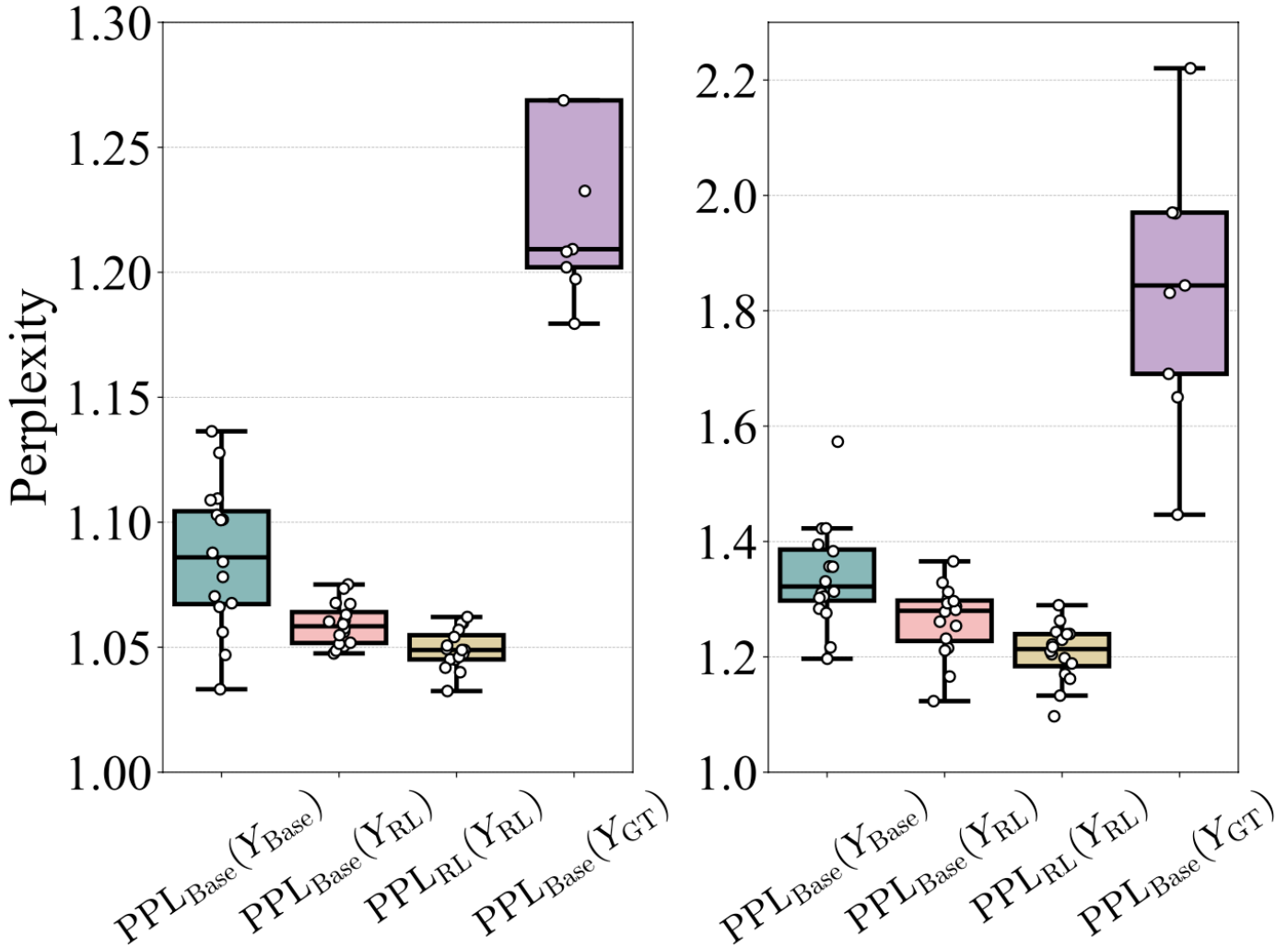
Deep Analysis

使用困惑度

$\mathbf{Y}_{\text{Base}}$（基模型生成的响应）

$\mathbf{Y}_{\text{RL}}$（RL 模型生成的响应）

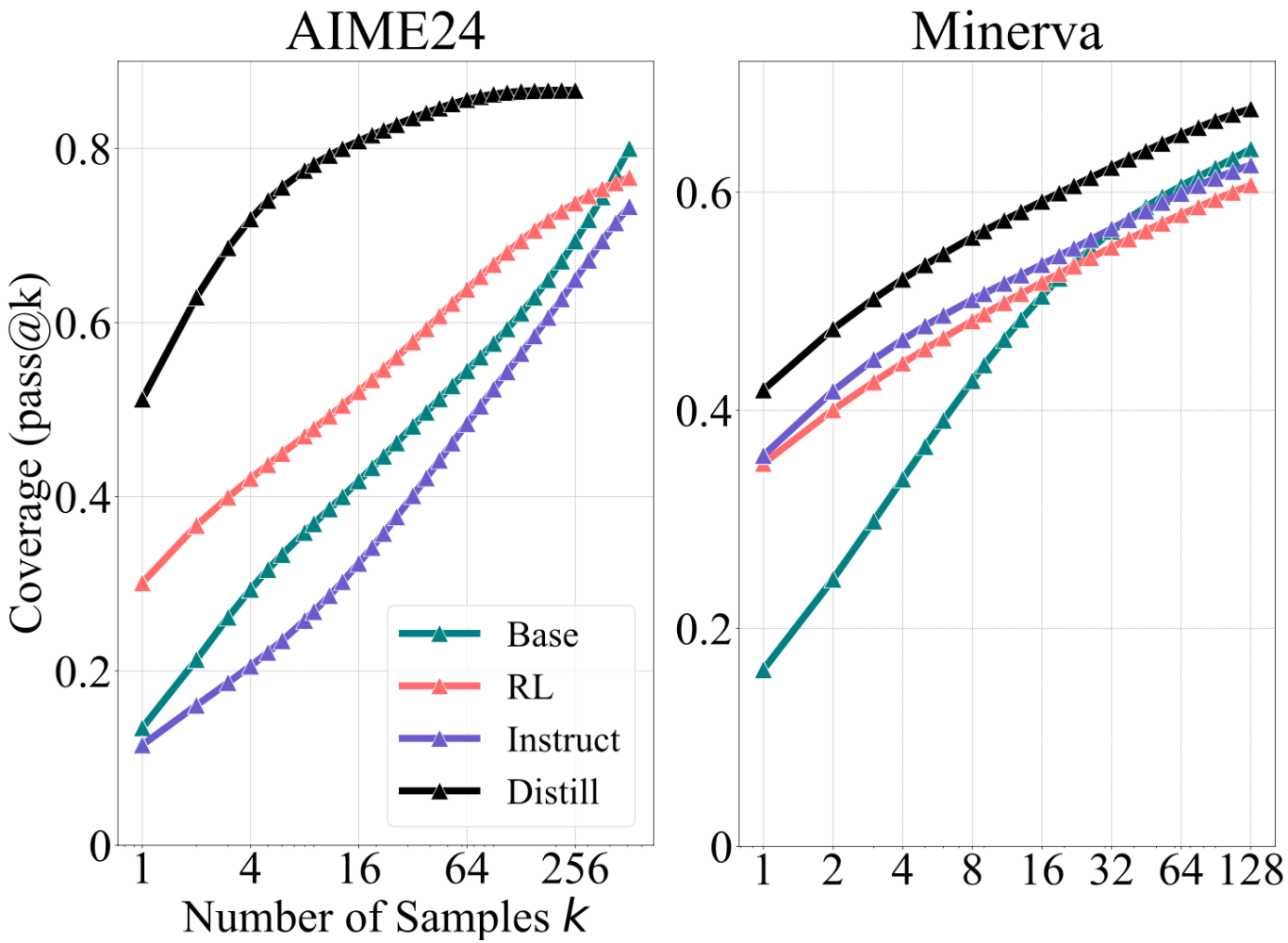
$\mathbf{Y}_{\text{GPT}}$（第三方模型生成的响应）



$\text{PPL}_{\text{Base}}(\mathbf{Y}_{\text{RL}}|x)$ 的分布非常接近于 $\text{PPL}_{\text{Base}}(\mathbf{Y}_{\text{Base}}|x)$ 的低困惑度部分

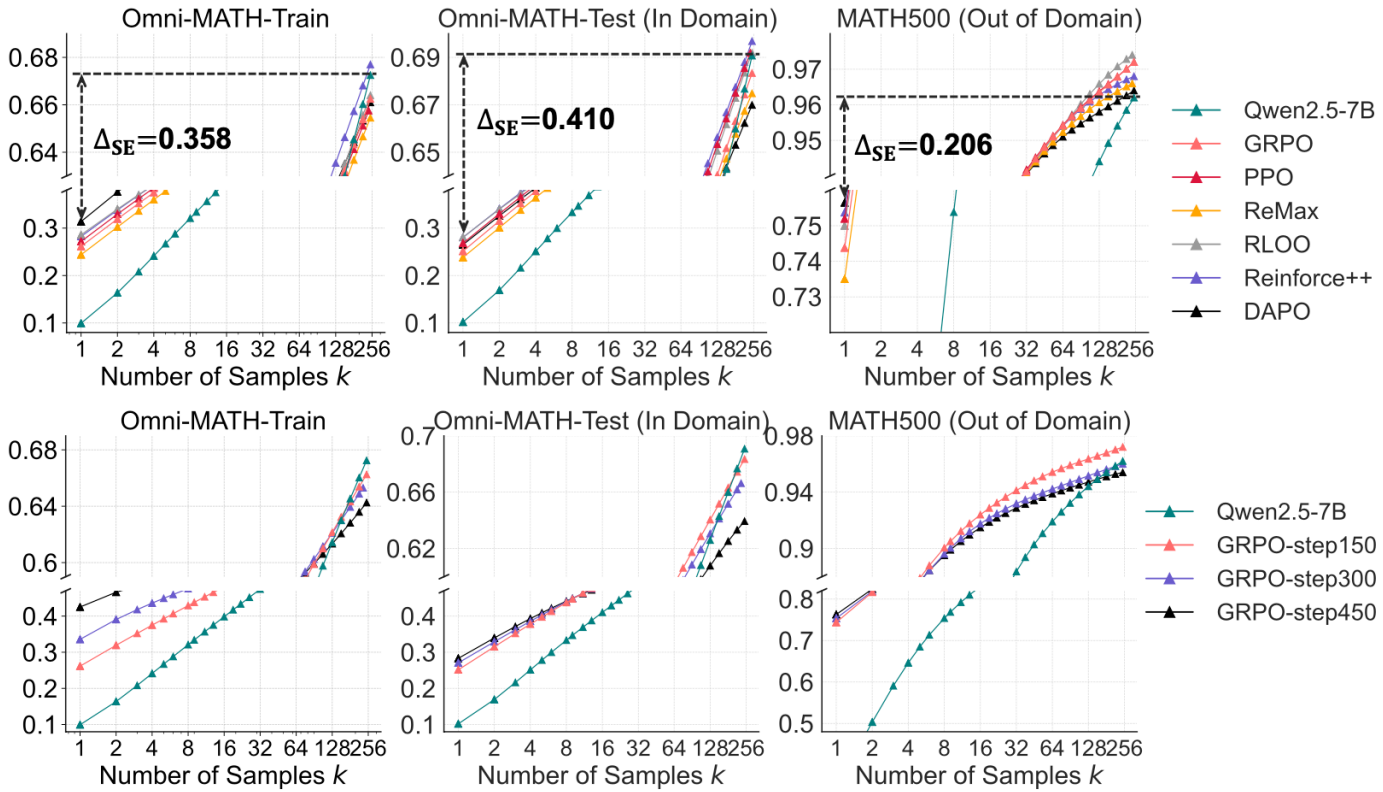
RL 模型生成的响应很可能也是基模型倾向生成的结果

Distillation Expands the Reasoning Boundary



这表明与受限于基模型推理能力的 RL 不同，蒸馏能够从更强的教师模型中引入新的推理模式，从而使蒸馏后的模型具备超越基模型推理边界的能力

Effects of Different RL Algorithms



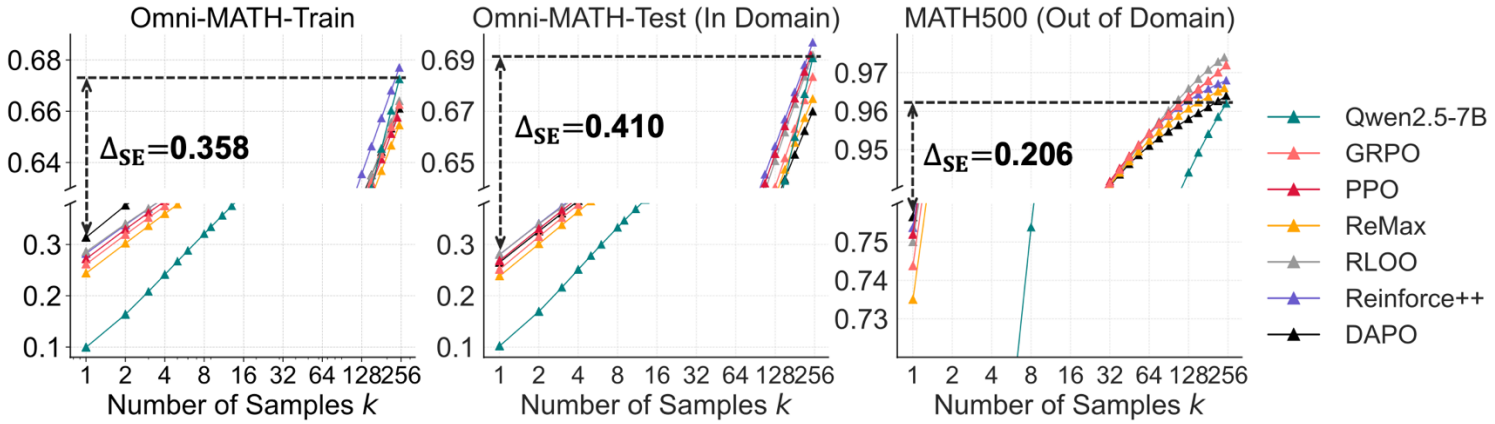
$$\Delta\text{SE} = \text{pass@1}_{\text{RL}} - \text{pass@256}_{\text{Base}}$$

- **ΔSE 越小**，RL 模型和基模型的差距小，从base到RL，**采样效率不算高**
- ΔSE 越大，RL 模型和基模型的差距大，从base到RL，**采样效率不算高**

不同的 RL 算法 ΔSE 都在 40 分以上，现有的 RL **方法**虽然**能**让模型更快“**答**对”，**但**远远没有达到最优采样效率

Effects of Different RL Algorithms

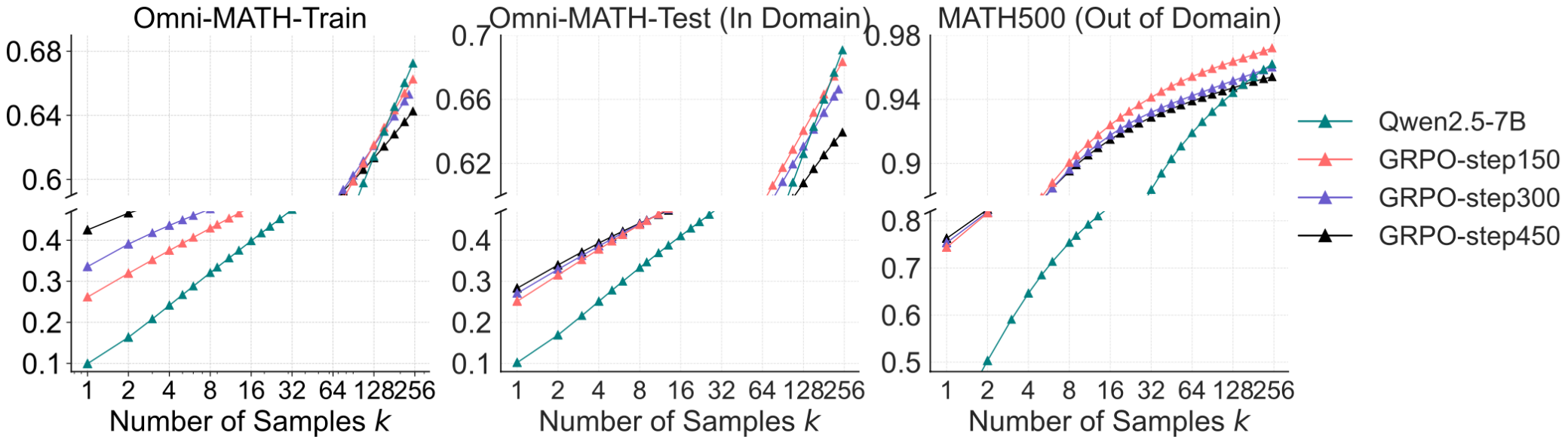
$$\Delta\mathrm{SE} = \mathrm{pass@1}_{\mathrm{RL}} - \mathrm{pass@256}_{\mathrm{Base}}$$



- **ΔSE 越小，** RL 模型和基模型的差距小，从base到RL，**采样效率不算高**
- ΔSE **越大，** RL 模型和基模型的差距大，从base到RL，**采样效率不算高**

不同的 RL **算法** ΔSE **都在** 40 **分以上，** 现有的 RL **方法**虽然**能**让模型更快**“答对”，但**远远没有达到最优采样效率

Asymptotic Effects of RL Training



- **在测试集上的收益边际递减**：**在域内与域外**测试集中，训练至第 450 步后的，远低于训练集上观察到的增幅。

- **pass@256 随训练步数增加而下降**：随着 RL 训练的深入，模型输出熵与探索能力逐渐下降所致。

Discussion

讨论一：传统强化学习与适用于大语言模型的 RLVR 的核心差异在于动作空间极其庞大及预训练先验的存在

传统 RL 与 LLM 场景下的 RLVR 存在两个本质性差异：

1. **动作空间呈指数级增长：**语言模型的动作空间远大于围棋或 Atari 游戏，而主流 RL 算法最初并未设计用于如此庞大的离散动作空间。在缺乏先验知识的情况下进行训练几乎无法有效探索奖励信号。
2. **预训练先验的引入：**RLVR 训练并非从零开始，而是基于一个已经预训练好的基模型进行。这种先验大幅**引导了模型生成合理响应，使探索**过程更容易，从而使策略更容易获得正向奖励反馈。

Discussion

讨论二：在庞大动作空间中，先验既是优势也是束缚

由于响应的采样过程受到预训练先验的强烈引导，RL 策略往往难以探索超出该先验的新型推理模式（任何偏离先验的响应都极可能是无效或无意义的输出，进而获得负向奖励）：

1. 策略梯度算法的目标**是最大化那些 在先验内且获得正奖励的响应，同时最小化那些 超出先验且获得负奖励的响应的似然**。结果就是，训练后的策略倾向于生成那些原本就在基模型先验中的响应，从而将推理能力限制在基模型的能力边界之内
2. **找到突破先验限制进行探索的方法**
3. Or 找到更好逼近base探索能力的？

北京大學
PEKING UNIVERSITY

TTRL: Test-Time Reinforcement Learning

Yuxin Zuo[*1] **Kaiyan Zhang**[*1] **Shang Qu**[1,2] **Li Sheng**[1,2] **Xuekai Zhu**[1]
Biqing Qi[2] **Youbang Sun**[1] **Ganqu Cui**[2] **Ning Ding**[†1,2] **Bowen Zhou**[†1,2]
[1]Tsinghua University [2]Shanghai AI Lab

https://github.com/PRIME-RL/TTRL

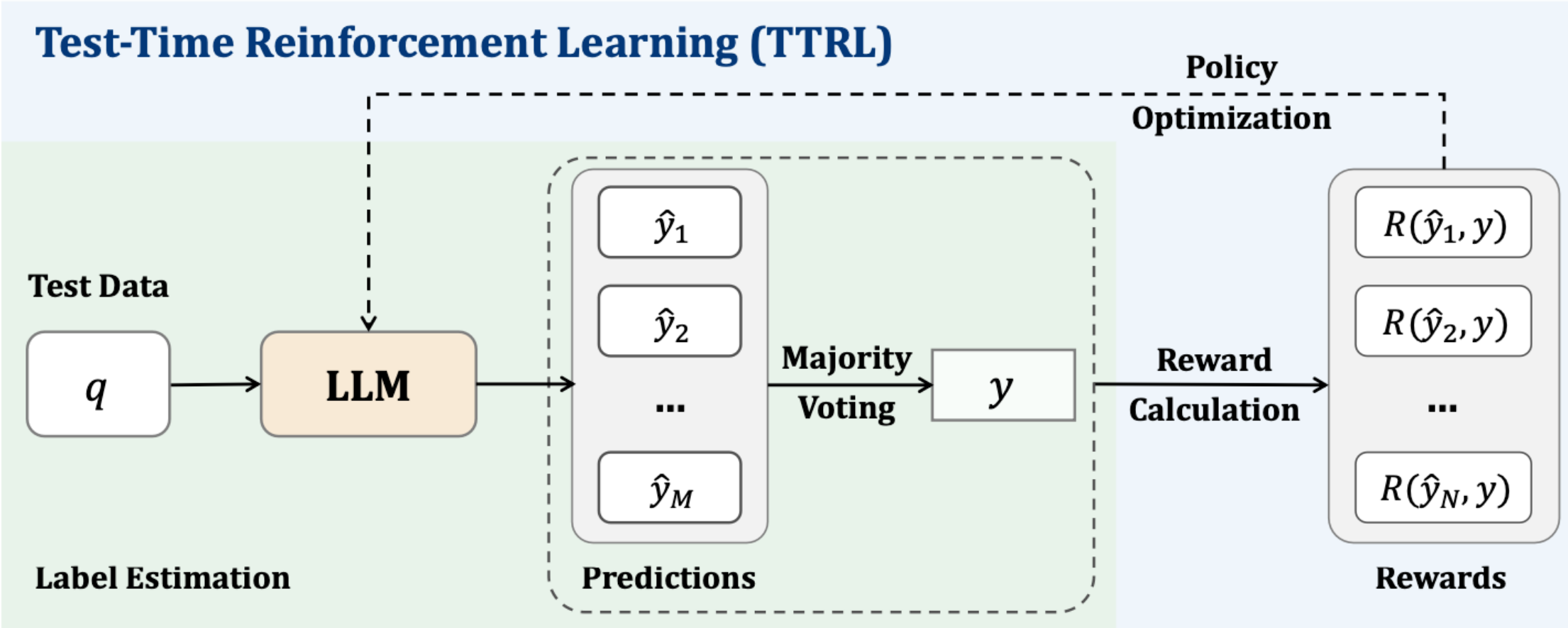


Figure 2: **TTRL** combines both Test-Time Scaling (TTS) and Test-Time Training (TTT).

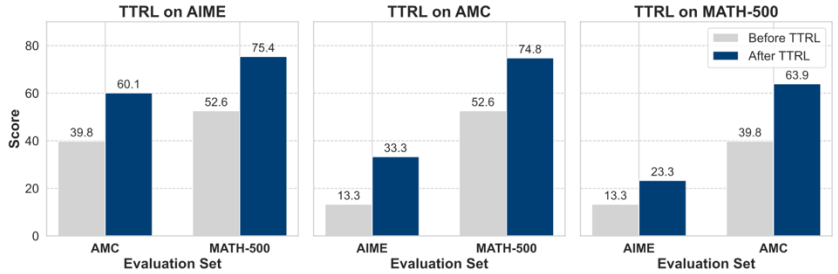
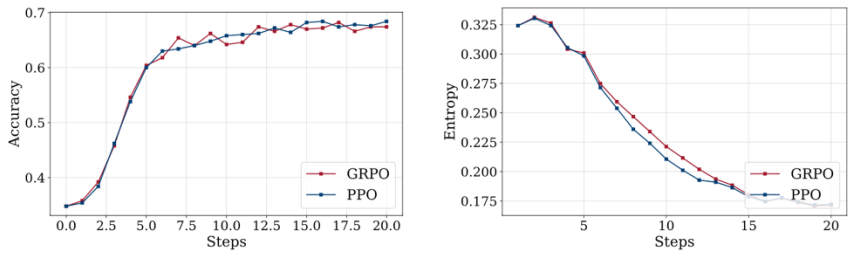
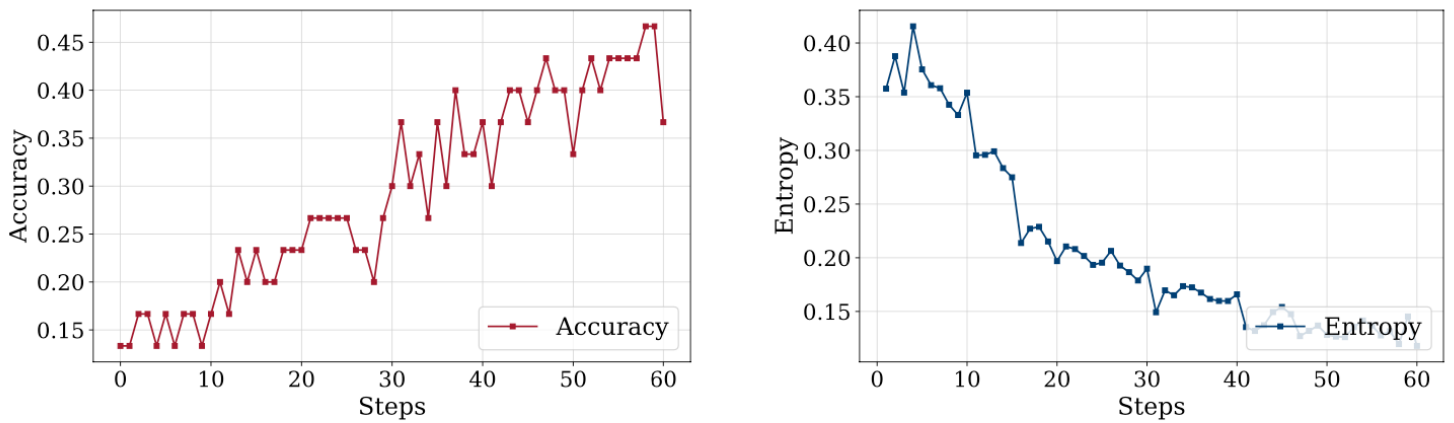


Figure 3: Out-of-distribution performance before and after **TTRL**.



(a) Accuracy Curve. (b) Entropy Curve.

Figure 4: Comparison over steps of different RL algorithms, GRPO *vs* PPO on MATH-500.



(a) Accuracy Curve. (b) Entropy Curve.

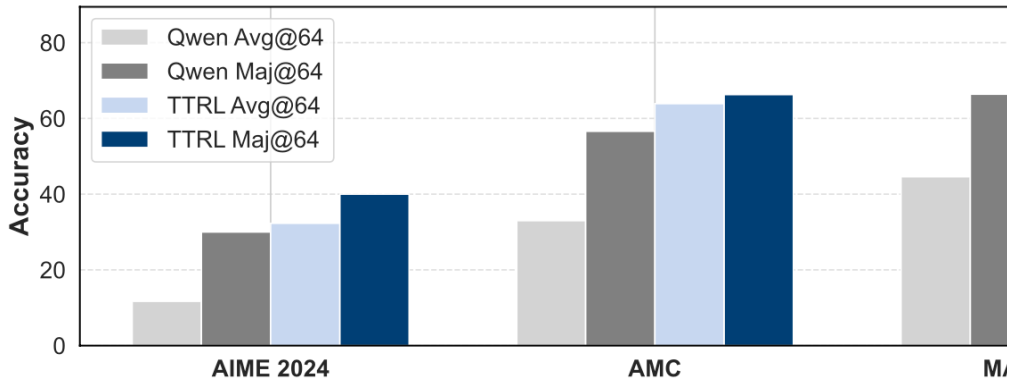


Figure 6: Majority voting performance comparison between backbone a

TTRL 由 Maj@N 监督训练，最终却超越了它。

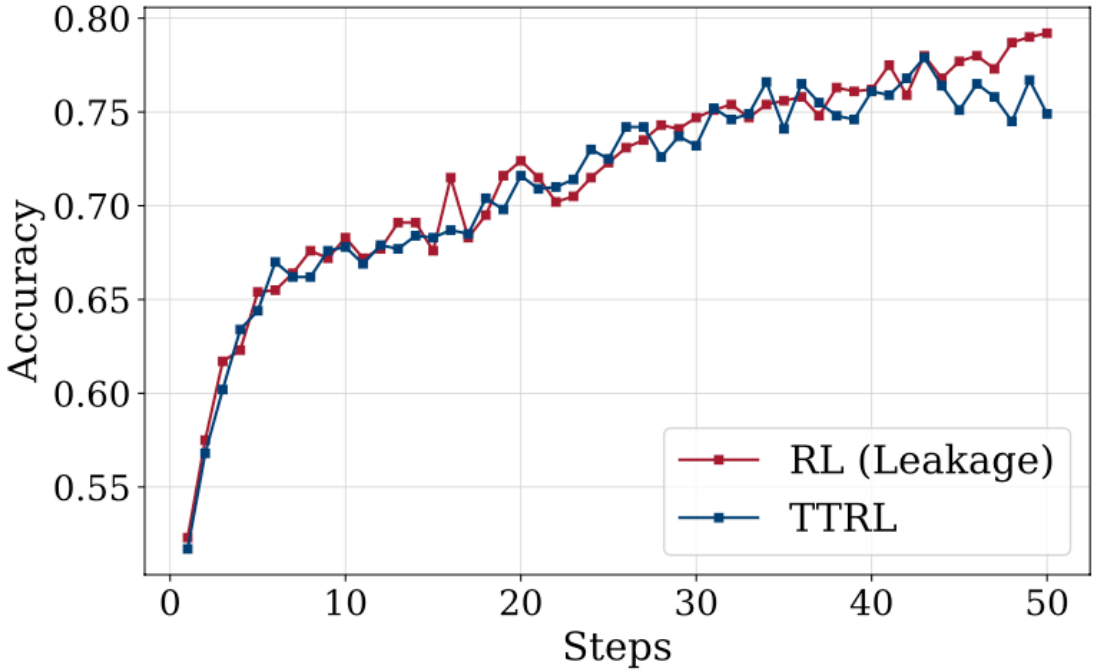


Figure 7: Comparison of RL (Leakage) *vs* **TTRL**.

TTRL 的性能提升接近于直接在基准测试集上训练所得的水平。

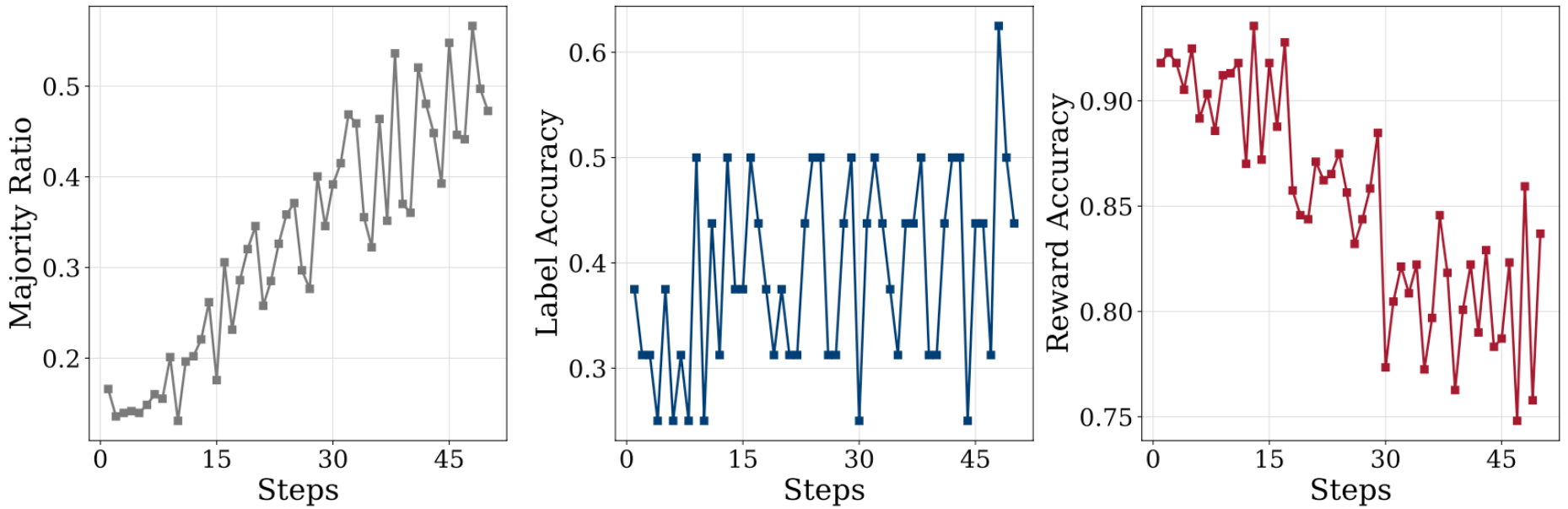


Figure 8: Comparison of Majority Ratio, Label Accuracy, and Reward Accuracy.

第一，奖励比标签更密集（dense），即使标签估计不准确，也能更频繁地获取有用的学习信号。例如，尽管**估**计标签是错误的，同一次 rollout **中的其他**输出仍可能带来正确或高质量的奖励，如图 9 **所示。**这使得整体奖**励信号**对伪标签误差具有更高的鲁棒性。

第二，一个有趣的现象是：模型能力越弱，TTRL 所给出的奖励反而可能更准确。

Table 2: Performance of **TTRL** across the five difficulty levels of MATH-500.

Metric	Name	MATH-500-L1	MATH-500-L2	MATH-500-L3	MATH-500-L4	MATH-500-L5
Accuracy	Backbone	25.9	33.0	36.3	32.5	22.3
	w/ **TTRL**	71.2	76.2	76.3	58.7	39.2
	Δ	+45.4 ↑ 175.3%	+43.2 ↑ 130.8%	+40.0 ↑ 110.2%	+26.2 ↑ 80.4%	+16.8 ↑ 75.3%
Response Len.	Backbone	2,339.2	2,125.1	2,120.6	1,775.1	1,751.3
	w/ **TTRL**	624.3	614.4	672.3	783.5	985.3
	Δ	−1,715.0 ↓ 73.3%	−1,510.6 ↓ 71.1%	−1,448.3 ↓ 68.3%	−991.6 ↓ 55.9%	−766.0 ↓ 43.7%

实验发现，随着问题难度的增加，TTRL 所带来的性能提升幅度以及解答长度缩减比率均**呈下降**趋势。这表明主干模型的可用先验知识不足，难以支撑在高难度题目上的学习过程

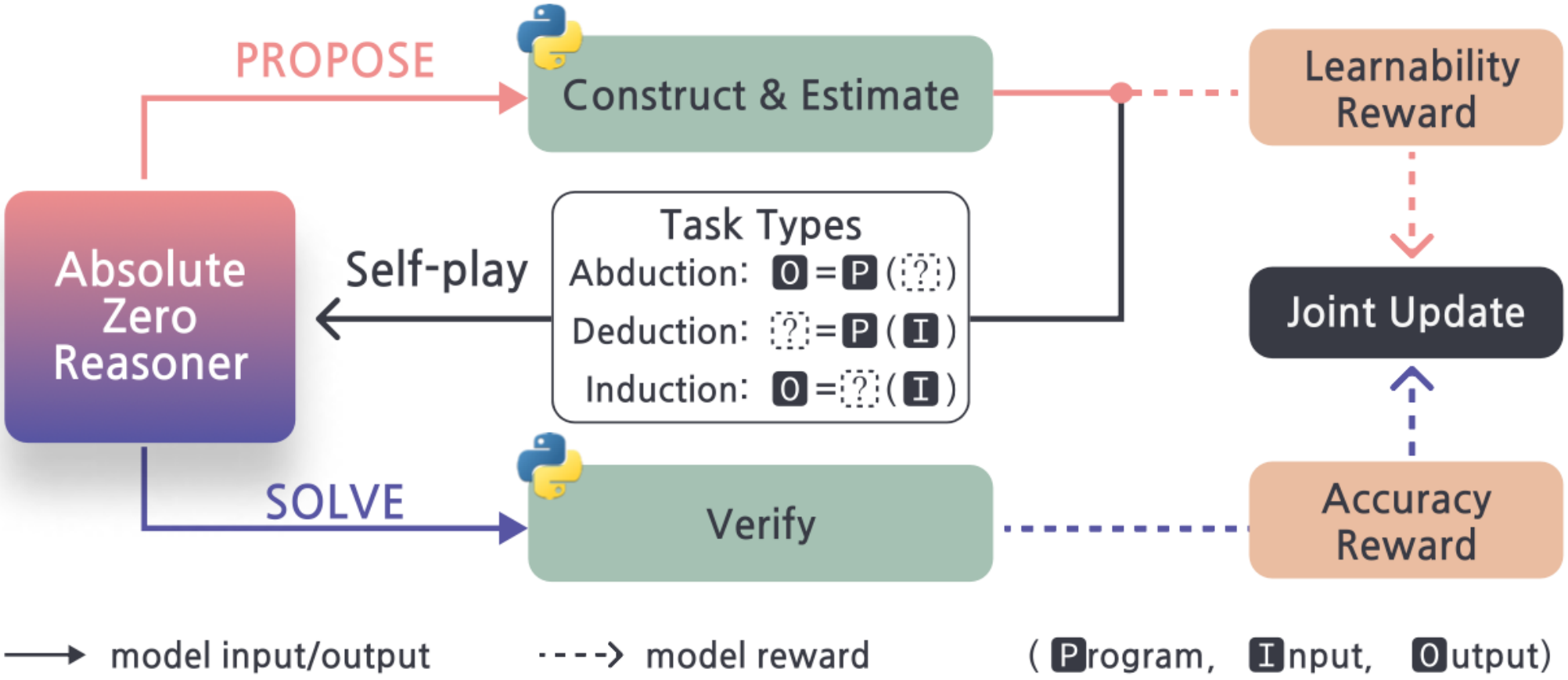
北京大學
PEKING UNIVERSITY

Absolute Zero: Reinforced Self-play Reasoning with Zero Data

Andrew Zhao [1], **Yiran Wu** [3], **Yang Yue** [1], **Tong Wu** [2], **Quentin Xu** [1], **Yang Yue** [1], **Matthieu Lin** [1], **Shenzhi Wang** [1], **Qingyun Wu** [3], **Zilong Zheng** [2,✉] **and Gao Huang** [1,✉]

[1] Tsinghua University [2] Beijing Institute for General Artificial Intelligence [3] Pennsylvania State University

zqc21@mails.tsinghua.edu.cn, yiran.wu@psu.edu, zlzheng@bigai.ai, gaohuang@tsinghua.edu.cn


PROPOSE
Construct & Estimate
Learnability Reward
Absolute Zero Reasoner
Self-play
Task Types
Abduction: O = P (?)
Deduction: ? = P (I)
Induction: O = ? (I)
Joint Update
SOLVE
Verify
Accuracy Reward
model input/output
model reward
(Program, Input, Output)

Program Triplet

Input: `"Hello World"`

```
1  def f(x):
2      return x
```

Output: `"Hello World"`

Model	Base	#data	HEval+	MBPP+	LCBv1-5	AME24	AME25	AMC	M500	Minva	Olypiad	CAvg	MAvg	AVG
Base Models														
Qwen2.5-7B[73]	-	-	73.2	65.3	17.5	6.7	3.3	37.5	64.8	25.0	27.7	52.0	27.5	39.8
Qwen2.5-7B-Ins[73]	-	-	75.0	68.5	25.5	13.3	6.7	52.5	76.4	35.7	37.6	56.3	37.0	46.7
Qwen2.5-7B-Coder[26]	-	-	80.5	69.3	19.9	6.7	3.3	40.0	54.0	17.3	21.9	56.6	23.9	40.2
Qwen2.5-7B-Math[74]	-	-	61.0	57.9	16.2	10.0	16.7	42.5	64.2	15.4	28.0	45.0	29.5	37.3
Zero-Style Reasoners Trained on Curated Coding Data														
AceCoder-RM[84]	Ins	22k	79.9	71.4	23.6	20.0	6.7	50.0	76.4	34.6	36.7	58.3	37.4	47.9
AceCoder-Rule[84]	Ins	22k	77.4	69.0	19.9	13.3	6.7	50.0	76.0	37.5	37.8	55.4	36.9	46.2
AceCoder-RM[84]	Coder	22k	78.0	66.4	27.5	13.3	3.3	27.5	62.6	29.4	29.0	57.3	27.5	42.4
AceCoder-Rule[84]	Coder	22k	80.5	70.4	29.0	6.7	6.7	40.0	62.8	27.6	27.4	60.0	28.5	44.3
CodeR1-LC2k[36]	Ins	2k	81.7	71.7	28.1	13.3	10.0	45.0	75.0	33.5	36.7	60.5	35.6	48.0
CodeR1-12k[36]	Ins	12k	81.1	73.5	29.3	13.3	3.3	37.5	74.0	35.7	36.9	61.3	33.5	47.4
Zero-Style Reasoners Trained on Curated Math Data														
PRIME-Zero[9]	Coder	484k	49.4	51.1	11.0	23.3	23.3	67.5	81.2	37.9	41.8	37.2	**45.8**	41.5
SimpleRL-Zoo[85]	Base	8.5k	73.2	63.2	25.6	16.7	3.3	57.5	77.0	35.7	41.0	54.0	38.5	46.3
Oat-Zero[38]	Math	8.5k	62.2	59.0	15.2	30.0	16.7	62.5	80.0	34.9	41.6	45.5	44.3	44.9
ORZ[23]	Base	57k	80.5	64.3	22.0	13.3	16.7	60.0	81.8	32.7	45.0	55.6	41.6	48.6
Absolute Zero Training w/ No Curated Data (Ours)														
AZR (Ours)	Base	**0**	71.3$^{-1.9}$	69.1$^{+3.8}$	25.3$^{+7.8}$	13.3$^{+6.6}$	13.3$^{+10.0}$	52.5$^{+15.0}$	74.4$^{+9.6}$	38.2$^{+13.2}$	38.5$^{+10.8}$	55.2$^{+3.2}$	38.4$^{+10.9}$	46.8$^{+7.0}$
AZR (Ours)	Coder	**0**	83.5$^{+3.0}$	69.6$^{+0.3}$	31.7$^{+11.8}$	20.0$^{+13.3}$	10.0$^{+6.7}$	57.5$^{+17.5}$	72.6$^{+22.6}$	36.4$^{+19.1}$	38.2$^{+16.3}$	**61.6**$^{+5.0}$	39.1$^{+15.2}$	**50.4**$^{+10.2}$

Table 1. **Performance of RL-Trained Reasoner on Reasoning Benchmarks Based on Qwen2.5-7B Models.** Performance of various models is evaluated on three standard code benchmarks (HumanEval+, MBPP+, LCBv1-5 and six math benchmarks (AIME'24, AIME'25, AMC'23, MATH500, Minerva, OlympiadBench). Average performance across coding and math benchmarks is calculated as average of the two averages: AVG = (CAvg + MAvg)/2. We use + for absolute percentage increase from base model. All models are trained using different variants of the `Qwen2.5-7B` model, with the variant and data usage labeled, more details listed in Table 4