北京大学深圳研究生院
Peking University Shenzhen Graduate School

Self-Distilled Reasoner: On-Policy Self-Distillation for Large Language Models

Meta **OPSD:** https://arxiv.org/abs/2601.18734
ERL： Experiential Reinforcement Learning
SDPO: https://arxiv.org/pdf/2601.20802

汪静雅

2025.03.18

01 Abstruct

研究背景: 知识蒸馏是提升小模型推理能力的有效手段, On-policy distillation(OPD)通过让学生对其自己的轨迹进行采样而教师LLM提供**密集的令牌级监督**来改进这种方法，从而解决了Off-policy distillation中训练和推理之间的**分布失配**。然而，**OPD通常需要单独的，通常更大的教师LLM，**并且没有明确地利用推理数据集中可用的Ground Truth解决方案。

研究重点:

- 提出假设：模型在“已知答案”时的反推能力要远强于“未知答案”时的生成能力，受直觉的启发，**一个足够有能力的LLM可以合理化外部推理痕迹，并教导其较弱的自我。**
- 提出**自蒸馏策略（OPSD）：一个模型通过不同的 Context 同时扮演两个角色**，让模型在自己的采样轨迹上，学习“看答案的自己”是如何思考的。

研究成果:

- 在多个数学推理基准上证明了方法的有效性，与GRPO等强化学习方法相比，实现了4-8倍的token效率，并且明显强于Off-policy方法。

后训练阶段的三种方法：具有可验证奖励的强化学习（RLVR），监督微调（SFT）或知识蒸馏（KD）。

- **RLVR**
  - 贵且不稳定：为了估算一个动作的好坏，需要针对同一个问题采样多次，计算成本高且方差大。
  - 梯度消失：若采样的结果全对或者全错，模型就学不到东西了（gradient signal vanishes）。
  - 反馈粗糙：它的奖励是 Outcome-based（结果导向）的，各步骤奖励很大程度上取决于结果正确性，忽略了 token-level的细粒度反馈。
- **SFT**
  - 暴露偏差：训练时老师根据ground truth手把手教，测试时模型得自己走。这种差异导致泛化能力较弱。
- **传统KD**
  - Off-policy：通常是用老师生成的数据来教学生，类似于让学生背老师的笔记，而不是让学生自己做题老师来改，将会导致 Distribution Mismatch（分布不匹配），且教师模型成本高。

→ **OPD:** 学生模型对其自己的轨迹进行采样，而教师政策提供密集的令牌级监督，将On-Policy训练与密集反馈相结合，显示出强大的性能。

OPD: 虽然性能强大，**但它依赖于一个独特的教师模型来监督学生。**

提出问题 ↓

"Can a model effectively serve as its own teacher through self-distillation?"

(一个模型能不能通过自我蒸馏，有效地充当自己的老师？)

灵感来源：当一个学生做错题时，让他看着正确答案去合理化解题步骤，比让他从头再做一遍要容易得多。且先前的工作证明，对于LLM，评估通常比生成更容易。

作出假设 ↓

Rationalization (解释给定的正确答案) 也比 Generation (直接生成答案) 要容易。

提出框架 ↓

OPSD： 让同一个模型，分饰两角：**Teacher Policy** 能看到 Privileged Information (特权信息)，负责看着答案写过程。**Student Policy** 只能看到题目，负责盲写。**训练目标即让**学生在自己生成的路径上，去模仿那个已知答案的自己的概率分布，即最小化两者在 token 级别的差异。

知识蒸馏 KD：知识蒸馏通过训练学生模仿教师的行为，将知识从较大的教师模型转移到较小的学生模型。核心观点是，教师输出软概率分布比单独的硬标签包含更丰富的信息。

- **传统的监督蒸馏：**目标是最小化在固定数据集上教师和学生分布之间的分歧D，差异项D衡量的是每一个 token 位置上的不一致。这样会导致distribution mismatch，即学生推理时与在固定数据集上（教师）训练时看到的知识不一致。

$$\mathcal{L}_{\text{Supervised Distillation}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{S}}[D(p_T\|p_S)(y|x)],\ D(p_T\|p_S)(y|x) = \frac{1}{|y|}\sum_{n=1}^{|y|} D(p_T(\cdot|y_{<n},x)\|p_S(\cdot|y_{<n},x))$$

- **OPD：**通过在其自己生成的序列上训练学生来解决这个分布不匹配的问题，从教师模型获得关于自己输出的密集令牌级反馈。这种方法将强化学习的政策相关性与监督学习的密集奖励信号相结合，从而在保持计算效率的同时减轻暴露偏差。

$$\mathcal{L}_{\text{On-Policy Distillation}}(\theta) = \mathbb{E}_{x\sim\mathcal{S}}[\mathbb{E}_{\hat{y}\sim p_S(\cdot|x)}[D(p_T\|p_S)(\hat{y}|x)]].$$

带验证奖励的强化学习RLVR：这类方法特别适用于具有易于验证结果的任务（如数学和编码）。首先对于每个问题，让模型生成一组回答；其次，根据结果对每个回答进行二值奖励。最后，计算优势（GRPO 用组内标准化的方式来计算优势），并基于这个优势去更新策略。

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{S} \\ o_1,\ldots,o_G\sim\pi_\theta(\cdot|x)}} \left[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|}\sum_{n=1}^{|o_i|} \min\left(\rho_i^n A_i, \text{clip}\left(\rho_i^n, 1-\varepsilon, 1+\varepsilon\right) A_i\right) - \beta D_{\text{KL}}[\pi_\theta(\cdot|x)\|\pi_{\text{ref}}(\cdot|x)] \right]$$

$$A_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^{G})}{\text{std}(\{r_j\}_{j=1}^{G})}.$$

虽然RLVR方法已经证明了强大的经验性能，但它面临两个关键限制：

（1）奖励信号是稀疏的，仅提供序列级反馈，而不是关于错误发生的令牌级指导，

（2）当所有采样的响应都收到相同的奖励（全部正确或全部不正确）时，优势变为零，尽管采样的计算成本很高，但仍无法进行任何策略更新。

可验证推理数据集

SFT处理：可以被视为使用专家轨迹的离线策略蒸馏/模仿学习，但它存在训练和推理之间的**分布不匹配**。

RLVR处理：通过比较答案与输出获得奖励来优化策略，但在**计算上昂贵**，且**回报信号稀疏**。

可以训练过程**奖励模型**（PRM）以在RL期间提供密集反馈。但训练PRM非常昂贵且难以扩展。

OPD处理：通过在学生自己的样本上训练，但是**需要单独的、更大的教师模型**来提供监督。

综上所述，我们寻求一种**密集的、On-Policy的、不需要外部教师或奖励模型**的OPSD方法。

	SFT/Off-Policy Distillation	GRPO	On-Policy Distillation	On-Policy Self-Distillation (Ours)
On-Policy Data	✗	✓	✓	✓
Dense Learning Signal	✓	✗	✓	✓
Low Sampling Cost	✓	✗	✓	✓
No External Teacher	✓	✓	✗	✓

On-Policy Self-Distillation

- **Motivation: Learning by understanding solutions.**
  - 根据学生的学习方式可以理解，当与问题作斗争时，比起一味试错，学生会选择查看答案，理解推理，并内化方法。同样，如果模型可以查询正确的答案或推理，并且有足够的能力，它也可以合理化出解题步骤。因此在OPSD的训练过程中，将直接利用地面真解y*作为特权信息，使得模型可以充当自己的老师，而不需要外部奖励模型或更大的老师模型。

- **Teacher and student policies.**
  - 通过改变条件上下文，从同一个语言模型pθ中实例化两个条件分布，这两种策略共享相同的参数。

Student Prompt

Problem: Find the derivative of $f(x) = 3x^2 + 2x - 5$ at $x = 2$

Answer:

$$p_S(\cdot \mid x) \triangleq p_\theta(\cdot \mid x).$$

学生策略只得到问题x，与实际推理条件匹配

Teacher Prompt

Problem: Find the derivative of $f(x) = 3x^2 + 2x - 5$ at $x = 2$

Here is a reference solution:
First find $f'(x) = 6x + 2$, then evaluate at $x = 2$: $f'(2) = 6(2) + 2 = 14$

After understanding the reference solution, please try to solve this problem using your own approach below:

Answer:

$$p_T(\cdot \mid x, y^\star) \triangleq p_\theta(\cdot \mid x, y^\star).$$

教师政策的条件是问题x和参考解决方案y*，并且被要求在合理化后参考答案后生成一个新的解决方案。

北京大学 信息工程学院
PEKING UNIVERSITY School of Electronic and Computer Engineering

On-Policy Self-Distillation

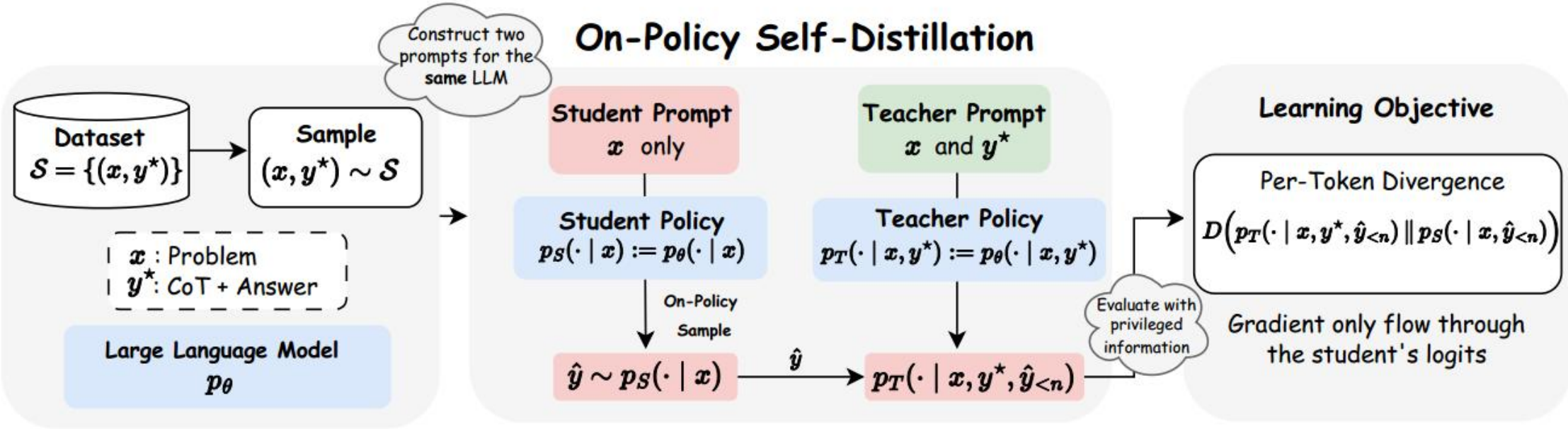


Figure 1. **Overview of On-Policy Self-Distillation (OPSD):** Given a reasoning dataset $\mathcal{S} = \{(x_i, y_i^\star)\}_{i=1}^N$, we instantiate two policies from the same LLM: a *student policy* $p_S(\cdot \mid x)$ and a *teacher policy* $p_T(\cdot \mid x, y^\star)$. The student generates an on-policy response $\hat{y} \sim p_S(\cdot \mid x)$. Both policies then evaluate this trajectory to produce next-token distributions $p_S(\cdot \mid x, \hat{y}_{<n})$ and $p_T(\cdot \mid x, y^\star, \hat{y}_{<n})$ at each step n. The learning objective minimizes the per-token divergence $D(p_T \| p_S)$ along the student's rollout. Crucially, gradients backpropagate only through the student's logits, allowing the model to self-distil.

On-Policy Self-Distillation

- **On-policy sampling from the student.**
  1. 给定一个问题x，学生生成一个符合策略的响应; $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_{|\hat{y}|}) \sim p_S(\cdot \mid x).$
  2. 对于学生生成的每一个token位置n，让Teacher和Student分别计算在前n-1个学生token的情况下，生成下一个token的概率分布。

$$p_S(y_n \mid x, \hat{y}_{<n}), \qquad p_T(y_n \mid x, y^\star, \hat{y}_{<n}), \quad \text{where } \hat{y}_{<n} \triangleq (\hat{y}_1, \ldots, \hat{y}_{n-1}).$$

对于**学生**而言：我走到这一步，下一步该怎么走?

对于**教师**而言：**看着正确答案y***，既然学生已经走到了这一步，下一步往哪走才是对的?

- **Training objective.**
  - 计算学生和教师分布之间的每个令牌的差异，并在学生自己的输出上将其最小化。在反向传播时，**梯度只通过学生策略传播**，教师策略虽然共享参数，但在这里被视为一个 固定目标。也就是说，OPSD不希望 Teacher 为了迁就 Student 而改变自己的判断，只希望 Student 努力去模仿 Teacher。

On-Policy Self-Distillation

Algorithm 1 On-Policy Self-Distillation (OPSD)

Require: Reasoning dataset $\mathcal{S} = \{(x_i, y_i^\star)\}_{i=1}^{N}$; language model p_θ; divergence D (e.g., JSD_β)

Define student policy $p_S(\cdot \mid x) := p_\theta(\cdot \mid x)$

Define teacher policy $p_T(\cdot \mid x, y^\star) := p_\theta(\cdot \mid x, y^\star)$ ▷ same parameters; different conditioning

while not converged **do**

  Sample a minibatch $\mathcal{B} \subset \mathcal{S}$

  for all $(x, y^\star) \in \mathcal{B}$ **do**

    Sample on-policy response $\hat{y} \sim p_S(\cdot \mid x)$

    Compute the token-wise divergence along the student rollout:

$$\ell(x, y^\star) \leftarrow D\big(p_T \,\|\, p_S\big)(\hat{y} \mid x) = \frac{1}{|\hat{y}|} \sum_{n=1}^{|\hat{y}|} D\big(p_T(\cdot \mid \hat{y}_{<n}, x, y^\star) \,\big\|\, p_S(\cdot \mid \hat{y}_{<n}, x)\big)$$

  Batch loss $\mathcal{L}_{\mathrm{OPSD}}(\theta) \leftarrow \frac{1}{|\mathcal{B}|} \sum_{(x,y^\star)\in\mathcal{B}} \ell(x, y^\star)$

  Update $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\mathrm{OPSD}}(\theta)$

Return trained parameters θ for inference-time policy $p_S(\cdot \mid x)$

On-Policy Self-Distillation

- **Training objective.**

$$\mathcal{L}_{\mathrm{OPSD}}(\theta) = \mathbb{E}_{(x,y^\star)\sim\mathcal{S}}\, \mathbb{E}_{\hat{y}\sim p_S(\cdot|x)} \sum_{n=1}^{|\hat{y}|} D\Big(p_T(\cdot \mid x, y^\star, \hat{y}_{<n}) \;\Big\|\; p_S(\cdot \mid x, \hat{y}_{<n})\Big).$$

- **Full-vocabulary divergence.全词表差异**
- 此处的D采用广义JSD散度测量，这种完整的词表提供了密集的token级反馈，教师根据y*的信息，将学生暴露在下一个可能的token上的整个分布中，并引导学生走向产生正确答案的推理路径。

$$D(p_T \,\|\, p_S)(\hat{y} \mid x) \triangleq \frac{1}{|\hat{y}|} \sum_{n=1}^{|\hat{y}|} D\Big(p_T(\cdot \mid x, y^\star, \hat{y}_{<n}) \,\Big\|\, p_S(\cdot \mid x, \hat{y}_{<n})\Big),\ \mathrm{JSD}_\beta(p_T \| p_S) = \beta D_{KL}(p_T \| m) + (1-\beta) D_{KL}(p_S \| m)$$

- **Sampled-token Distillation.采样Token蒸馏**
- 根据最近的策略蒸馏方法，形成了一个采样令牌成形信号，并使用策略梯度进行优化。

$$A_n(x, \hat{y}) = \log p_T(\hat{y}_n \mid x, y^\star, \hat{y}_{<n}) - \log p_S(\hat{y}_n \mid x, \hat{y}_{<n}),$$

- 并据此优化目标

$$\mathcal{L}(\theta) = -\mathbb{E}_{(x,y^\star)\sim\mathcal{S}}\left[\mathbb{E}_{\hat{y}\sim p_S(\cdot|x)}\left[\frac{1}{|\hat{y}|}\sum_{n=1}^{|\hat{y}|} A_n(x,\hat{y}) \times \log p_S(\hat{y}_n \mid x, \hat{y}_{<n})\right]\right].$$

On-Policy Self-Distillation

- **Training objective.**

$$\mathcal{L}_{\mathrm{OPSD}}(\theta) = \mathbb{E}_{(x,y^\star)\sim\mathcal{S}}\, \mathbb{E}_{\hat{y}\sim p_S(\cdot|x)} \sum_{n=1}^{|\hat{y}|} D\Big(p_T(\cdot \mid x, y^\star, \hat{y}_{<n}) \;\Big\|\; p_S(\cdot \mid x, \hat{y}_{<n})\Big).$$

  - **Full-vocabulary divergence.全词表差异**
  - **Sampled-token Distillation.采样Token蒸馏**
  - **二者对比：**
    - 采样Token蒸馏仅对采样的令牌进行操作，而无需在每一步显式地匹配全分布。这本质上是在衡量：对于选这个词，老师比学生自信多少？如果是正的，说明老师觉得这个词选得对，要鼓励；如果是负的，说明老师觉得不该选，要惩罚。
    - 然而，采样Token蒸馏只利用了采样到的那一个token信息，丢掉了 Teacher 对其他所有可能 token 的看法，所以**信息量（Dense feedback）要少得多**。

核心问题：

1. OPSD与SFT和GRPO在数学推理性能方面的比较如何？样本效率的改进是什么？
2. OPSD如何在不同的模型大小上扩展，自蒸馏是否需要更强大的模型能力？
3. 生成长度对训练性能和样本效率有何影响？
4. 与仅对采样令牌进行计算并通过策略梯度进行优化相比，计算全词表的logits差异是否有优势？

实验设置：

1. 模型：Qwen3-1.7B、Qwen 3-4B和Qwen 3-8B
2. 训练数据：OpenThoughts的数学推理子集，采样了多达30K个包含 Chain-of-Thought 推理过程的问题-答案对。
3. 基准：竞赛级别基准，包括AIME 2024，AIME 2025，HMMT 2025和Amo-Bench。

实施细节：

1. 对于GRPO，每个问题采样8个回答；对于OPSD，每个问题采样1个回答。
2. 重要的是，**训练中将教师策略固定为初始策略，而不会随着学生一起更新**。作者发现这有助于稳定训练，并隐式地充当正则化，以防止过度偏离初始策略。这就好比老师得有一个稳定的标准，不能学生每学一点，老师的标准就变一点。

北京大学 信息工程学院
PEKING UNIVERSITY School of Electronic and Computer Engineering

1. OPSD与SFT和GRPO在数学推理性能方面的比较如何？样本效率的改进是什么？

Method	AIME24	AIME25	HMMT25	AMO-Bench	Average
Qwen3-8B					
Base (Instruct)	75.2	68.3	43.1	13.4	50.0
+ SFT	76.3	66.2	44.7	12.9	50.0
+ GRPO	76.7	68.7	45.0	**14.8**	51.3
+ OPSD	**77.5**	**69.8**	**47.1**	14.3	**52.2**
Qwen3-4B					
Base (Instruct)	74.6	65.8	40.3	12.4	48.3
+ SFT	75.2	66.3	44.4	12.5	49.6
+ GRPO	75.6	**67.1**	42.7	12.8	49.6
+ OPSD	**76.0**	66.9	**45.8**	**13.5**	**50.6**
Qwen3-1.7B					
Base (Instruct)	50.2	35.2	25.4	4.3	28.8
+ SFT	48.3	36.3	23.3	3.9	28.0
+ GRPO	**52.1**	38.3	**26.7**	4.5	**30.5**
+ OPSD	51.4	**39.5**	25.8	**5.0**	30.4

- **SFT vs. Base**：SFT 相比 Base 模型有小幅度提升
- **OPSD vs. SFT**：OPSD 在所有规模上都击败了 SFT。这说明，让模型在自己的生成路径上学习（On-policy），确实比单纯模仿专家（Off-policy）要有效。
- **OPSD vs. GRPO**：在 4B 和 8B 的规模上，OPSD 匹配甚至超过了 GRPO 的表现。

1. OPSD与SFT和GRPO在数学推理性能方面的比较如何？样本效率的改进是什么？

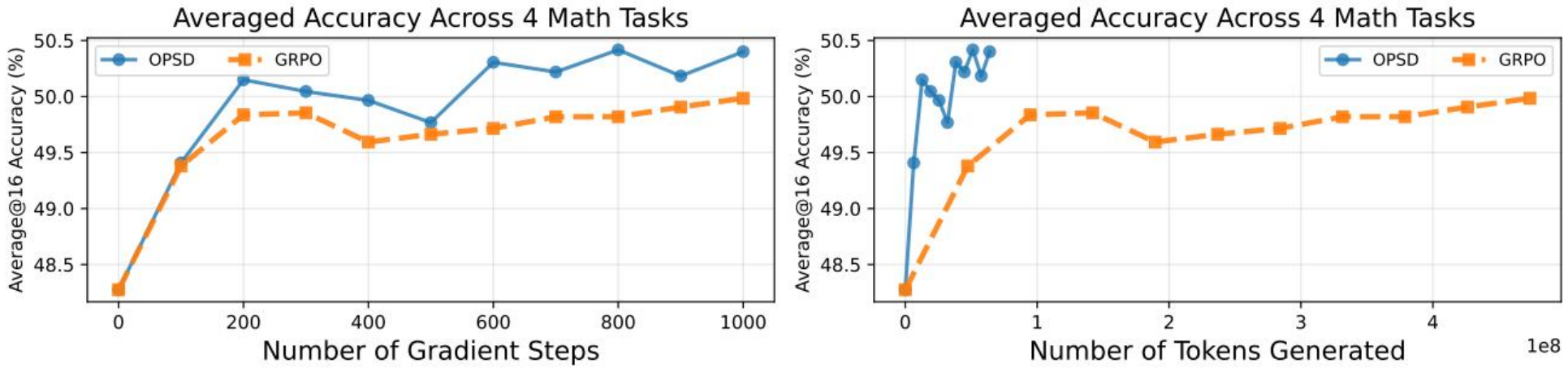


Figure 3. **Token Efficiency of OPSD.** We compare OPSD and GRPO on Qwen3-4B under the same effective training batch size, reporting average@16 performance as a function of gradient update steps and total generated tokens. Both methods are trained with the same effective batch size in terms of sampled generations per update, but differ in generation length: each generation is capped at 2048 tokens for OPSD and 16384 tokens for GRPO. OPSD achieves comparable or better performance with substantially fewer generated tokens, resulting in lower sampling cost and reduced training time. In this experiment, OPSD can be 4-8× more token-efficient than GRPO.

在相同的有效训练批次下，对于OPSD，每代的上限为2048个token，对于GRPO，每代的上限为16384个token。OPSD 在达到同样的性能的同时，生成的 token 数量比 GRPO **少了 4-8倍**。这种效率源于教师分布的**密集令牌级监督**，在不牺牲性能的情况下减少采样成本和训练时间。

2. OPSD如何在不同的模型大小上扩展，自蒸馏是否需要更强大的模型能力?

Method	AIME24	AIME25	HMMT25	AMO-Bench	Average
Qwen3-8B					
Base (Instruct)	75.2	68.3	43.1	13.4	50.0
+ SFT	76.3	66.2	44.7	12.9	50.0
+ GRPO	76.7	68.7	45.0	**14.8**	51.3
+ OPSD	**77.5**	**69.8**	**47.1**	14.3	**52.2**
Qwen3-4B					
Base (Instruct)	74.6	65.8	40.3	12.4	48.3
+ SFT	75.2	66.3	44.4	12.5	49.6
+ GRPO	75.6	**67.1**	42.7	12.8	49.6
+ OPSD	**76.0**	66.9	**45.8**	**13.5**	**50.6**
Qwen3-1.7B					
Base (Instruct)	50.2	35.2	25.4	4.3	28.8
+ SFT	48.3	36.3	23.3	3.9	28.0
+ GRPO	**52.1**	38.3	**26.7**	4.5	**30.5**
+ OPSD	51.4	**39.5**	25.8	**5.0**	30.4

OPSD依赖于教师在特权信息的条件下合理化参考解决方案的能力。在固定的数据集下，这种能力取决于足够的模型容量，并且预计会随着模型的大小而扩展。

因此，可以假设**OPSD随着模型越来越能够利用特权上下文而变得越来越有效。**

由结果可知，OPSD在1.7B下相比GRPO 提升有限，但在4B和8B标度下产生了逐渐更大的改善，这与假设一致。

3. 生成长度对训练性能和样本效率有何影响?

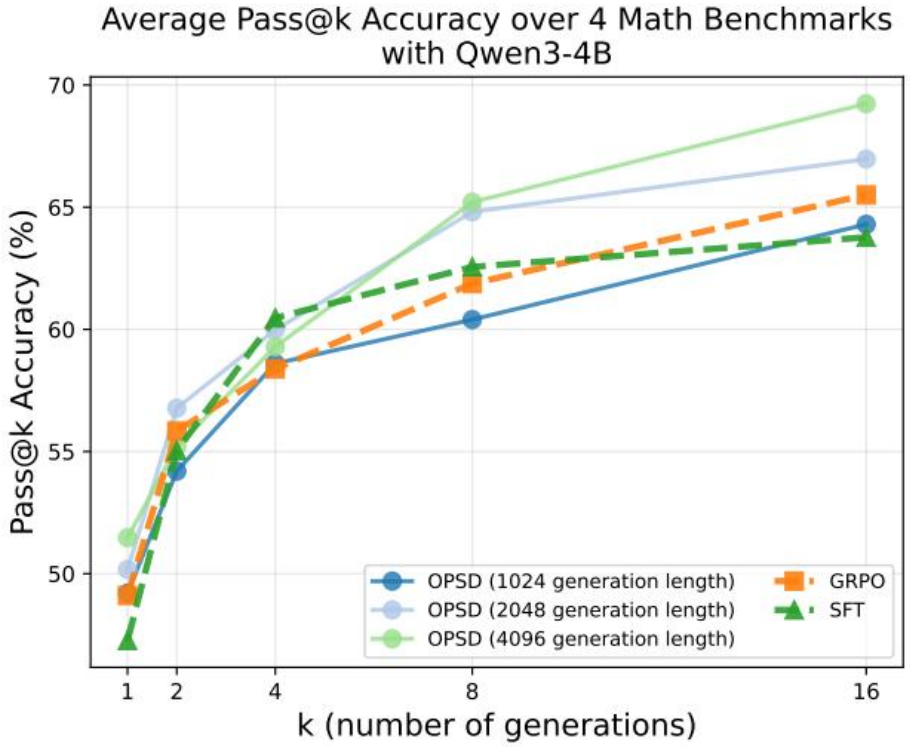


Figure 4. Pass@K performance averaged across four mathematical reasoning benchmarks for Qwen3-4B. We study the effect of the generation length of on-policy sampled student responses in OPSD, comparing 1024, 2048, and 4096 tokens. Longer generations provide more teacher signals. Increasing the generation length from 1k to 2k and 4k consistently improves pass@K, with both 2k and 4k substantially outperforming the 1k setting.

- 由于OPSD的目标是在令牌级别上操作，因此每个样本生成的令牌数量直接决定了学生可用的监督信号量。较长的序列使学生获得更多的教师反馈，但它们也增加了计算成本，并且可能引入噪声或无信息的延续。
- 为了研究这一权衡，作者以 Qwen3-4B 模型为对象进行了消融实验，改变在线采样的学生回答长度（1024、2048 和 4096 个 Token），并采用全词表 Logit 蒸馏。
- 由实验结果可知，增加生成长度会使pass@K性能的明显改善。特别是，2048-token和4096-token设置都显著优于1024-token基线，这**表明更长的学生输出有助于更有效的推理监督。**

4. 与仅对采样令牌进行计算并通过策略梯度进行优化相比，计算全词表的logits差异是否有优势?

Table 3. Ablation on divergence computation strategies for OPSD on Qwen3-4B with 2048 generation length for distillation. We report pass@8 accuracy on AIME25 and HMMT25. Full-distribution objectives (logit distillation) outperform sampled-token objectives.

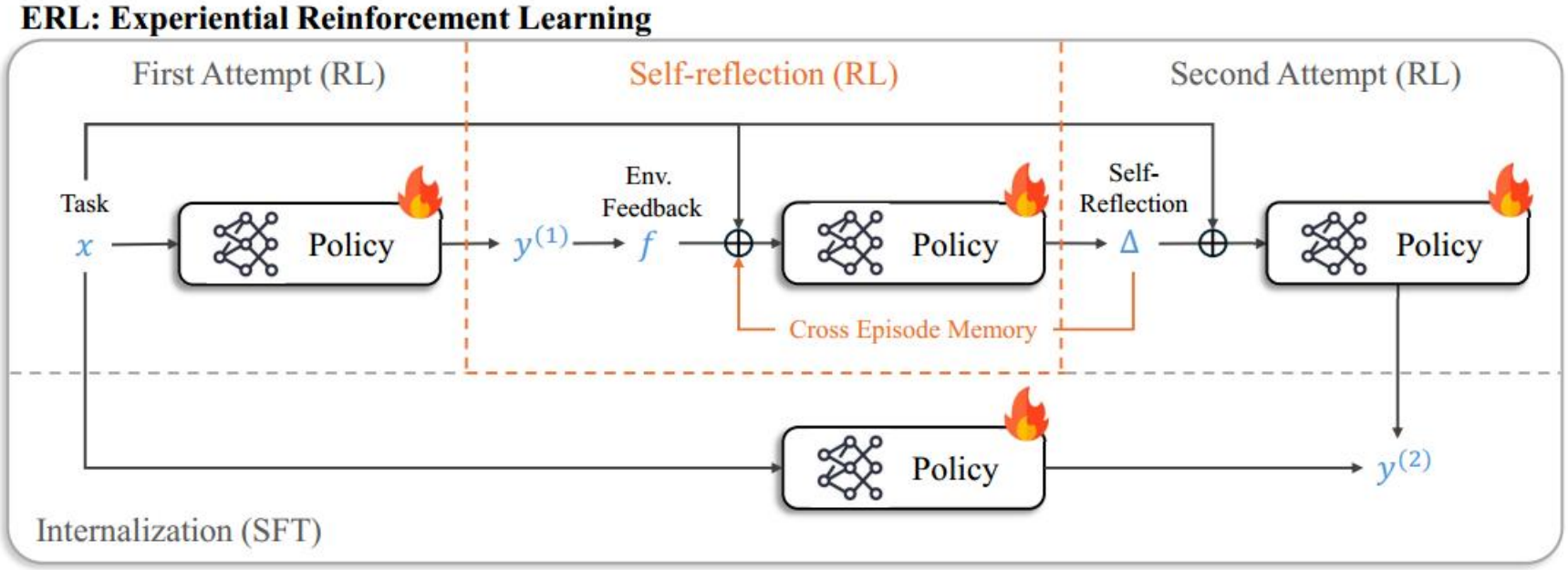
Method Variant	AIME25	HMMT25
OPSD w/ Full-vocabulary logit distillation (Agarwal et al., 2024)	**84.1**	**60.0**
OPSD w/ Sampled-token distillation (Lu & Lab, 2025)	82.1	57.3

- 全词表Logit蒸馏直接匹配完整的令牌分布，而采样令牌蒸馏由教师的对数概率形成训练目标。
- 由实验结果可知，**全词表Logit蒸馏显著超过采样令牌蒸馏**。
- 这是由于全词表包含了 Dark Knowledge (暗知识)。Teacher 不仅告诉了 Student 正确答案的概率是多少，还通过分布告诉了它哪些错误答案是“情有可原”的，哪些是“离谱”的。这种信息密度远高于只评价当前选的那个词。然而，全词汇计算会导致更高的显存使用，这表明**性能和效率之间的权衡。**

04 Limitations and Future Directions

- **模型规模：** 受限于计算资源，本文的实验仅针对参数量不超过8B的模型展开。尽管实验观察到更大规模的模型能从OPSD方法中获益更多，但在8B参数以上的规模（如70B或更前沿的大模型）下，这一趋势是否仍会延续，仍是一个待解决的开放问题。（但这样是否失去了“知识蒸馏”本身帮助小模型的意义？/大模型不需要人类标注数据，自己做题看答案反思即可不断进化？）
- **验证信号：** 我们当前框架**并未显式利用生成答案的正确性验证信号**；若将这类信号纳入学习过程，有望在分布匹配之外提供额外的优化目标。
- **问题难度：** 问题难度在自蒸馏（self-distillation）中扮演着关键角色，若推理问题的难度超出模型的理解阈值，即便教师策略能访问真实解，也无法提供有意义的监督信号。这表明，课程学习策略（curriculum learning strategies），即随模型能力提升逐步增加问题难度，能够提升训练有效性。 探索能**将问题维持在模型能力前沿**的自适应问题设计，是将 OPSD 方法拓展至更具挑战性的推理任务的重要方向。

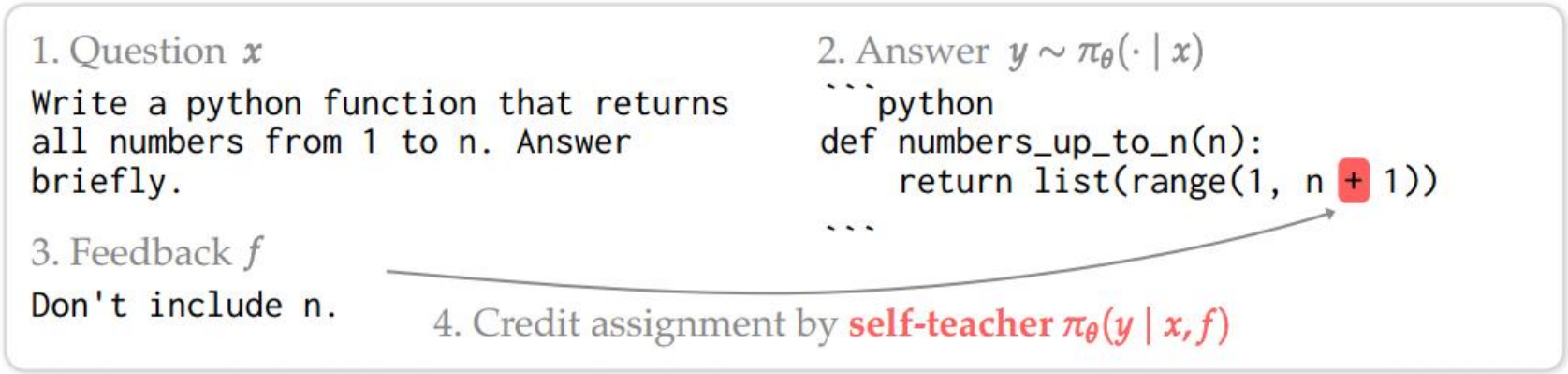
- **Experiential Reinforcement Learning：** Microsoft
- 给定输入任务x，语言模型首先产生初始尝试并接收环境反馈。当初次反馈不理想时，同一个模型将生成一个以这次尝试为条件的**自我反思**，用于指导**第二次尝试**。尝试和反思都通过强化学习进行优化，而成功的第二次尝试则通过自我升华（存储高质量反思进入记忆）进行内化，因此模型学习直接从原始输入模仿改进的行为，而无需自我反思。



Algorithm 1 Experiential Reinforcement Learning

Inputs: Language model π_θ; dataset of questions x; reward threshold τ; environment returning feedback f and reward r.
Initialize: reflection memory $m \leftarrow \emptyset$.
repeat
  Sample question x from the dataset.
  // First attempt
  Sample an answer $y^{(1)} \sim \pi_\theta(\cdot \mid x)$.
  Obtain environment feedback and reward $(f^{(1)}, r^{(1)})$.
  // Self-reflection
  Sample a reflection $\Delta \sim \pi_\theta(\cdot \mid x, y^{(1)}, f^{(1)}, r^{(1)}, m)$.
  // Second attempt
  Sample a refined answer $y^{(2)} \sim \pi_\theta(\cdot \mid x, \Delta)$.
  Obtain environment feedback and reward $(f^{(2)}, r^{(2)})$.
  Set reflection reward $\tilde{r} \leftarrow r^{(2)}$.
  Store reflection $m \leftarrow \Delta$ if $r^{(2)} > \tau$.
  // RL update
  Update θ via $\mathcal{L}_{\text{policy}}(\theta)$ over the first attempt, reflection, and second attempt.
  // Internalization
  Update θ via $\mathcal{L}_{\text{distill}}(\theta)$ to internalize reflection, training π_θ to produce $y^{(2)}$ from x only.
until converged

- **RL via Self-Distillation (SDPO)：** ETH Zurich
- student 基于问题生成回答，环境返回反馈；随后模型**再次前向**，在带**环境反馈**上下文的提示下得到self-teacher分布，只对原轨迹的 token log-prob 做重评估。



Algorithm 1 SDPO

Input: Language model π_θ; dataset with questions x; number of rollouts G per question; environment to obtain feedback for attempts.

repeat
  Sample question x from dataset.
  Sample responses: $\{y_i\}_{i=1}^{G} \sim \pi_\theta(\cdot \mid x)$.
  Evaluate responses to obtain feedback f_i.
  ▷ **Self-distillation:**
  Compute log-probs of self-teacher $\log \pi_\theta(y_{i,t} \mid x, f_i, y_{i,<t})$.
  Update θ with gradient descent on $\mathcal{L}_{\text{SDPO}}(\theta)$.
until converged

$$\mathcal{L}_{\text{SDPO}}(\theta) := \sum_t \text{KL}(\pi_\theta(\cdot \mid x, y_{<t}) \| \text{stopgrad}(\pi_\theta(\cdot \mid x, f, y_{<t})))$$

$$A_{i,t}^{\text{GRPO}} := r_i - \text{mean}\{r_i\}_{i=1}^{G} \text{ (constant in } t\text{)}, \quad A_{i,t}^{\text{SDPO}}(\hat{y}_{i,t}) = \log \frac{\pi_\theta(\hat{y}_{i,t} \mid x, f_i, y_{i,<t})}{\pi_\theta(\hat{y}_{i,t} \mid x, y_{i,<t})}.$$

$$A_n(x, \hat{y}) = \log p_T(\hat{y}_n \mid x, y^\star, \hat{y}_{<n}) - \log p_S(\hat{y}_n \mid x, \hat{y}_{<n}),$$

北京大学深圳研究生院
Peking University Shenzhen Graduate School

请批评指正！

OPSD: https://arxiv.org/abs/2601.18734
ERL： Experiential Reinforcement Learning
SDPO: https://arxiv.org/pdf/2601.20802

汪静雅

2026.03.18