Reinforcement Learning via Self-Distillation

Jonas Hübotter[1], Frederike Lübeck*,[1,2], Lejs Behric*,[1], Anton Baumann*,[1], Marco Bagatella[1,2], Daniel Marta[1], Ido Hakimi[1], Idan Shenfeld[3], Thomas Kleine Buening[1], Carlos Guestrin[4], Andreas Krause[1]

[1] ETH zürich [2] MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS [3] MIT [4] Stanford University — Learning & Adaptive Systems

RLVR has a signal and credit assignment bottleneck!

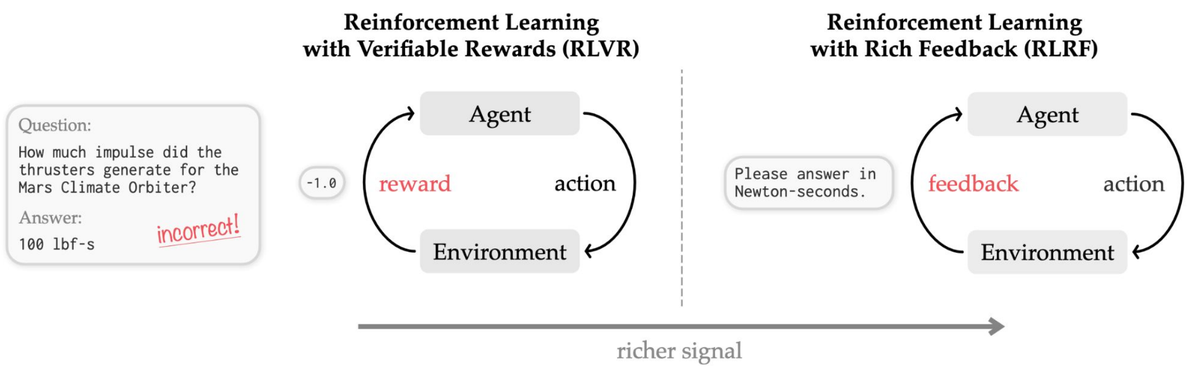
The solution: **using in-context learning for RL.**

Motivation

- RLVR with binary rewards receives 1 bit signal per rollout → **feedback bottleneck**.
- GRPO assigns the same advantage to each token → **credit assignment bottleneck**.

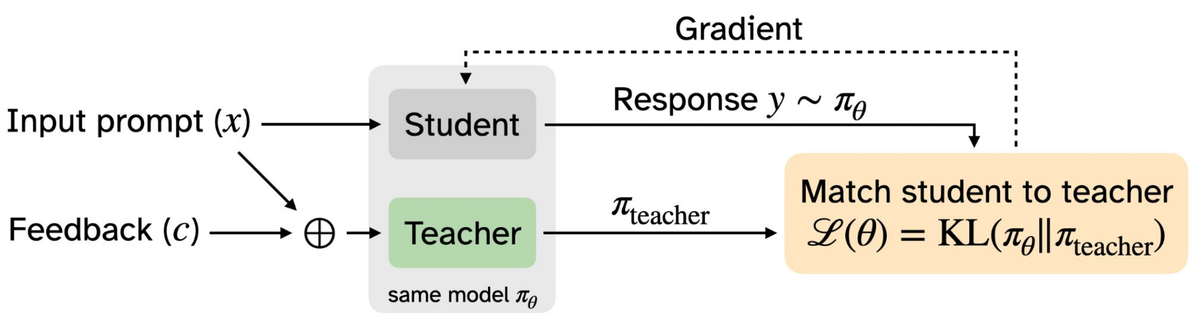
Observation: Models often understand *why* an attempt failed after reflection or rich feedback:

- Examples of rich feedback: runtime errors, failed unit tests, demonstrations, user feedback, …



Self-Distillation (**SDPO**)

"Leverage in-context learning for in-weight learning."



Two equivalent perspectives:

1. On-policy RL with teacher-based advantages

$$\text{Maximize} \quad A_i(x, y, c) := \log \frac{\pi_\theta(y_i \mid x, c, y_{<i})}{\pi_\theta(y_i \mid x, y_{<i})}$$

2. On-policy distillation with a context-based teacher

$$\text{Minimize} \quad \mathcal{L}_{\text{SDPO}}(\theta) := \frac{1}{|y|} \sum_{i=1}^{|y|} \mathrm{KL}(\pi_\theta(\cdot \mid x, y_{<i}) \| \pi_\theta(\cdot \mid x, c, y_{<i}))$$

Both perspectives yield same gradient in expectation!

An example with Qwen3-8B:

1. Question x

```
Write a python function that returns
all numbers from 1 to n. Answer
briefly.
```

2. Answer $y \sim \pi_\theta(\cdot \mid x)$

```
```python
def numbers_up_to_n(n):
    return list(range(1, n + 1))
```
```

3. Feedback f

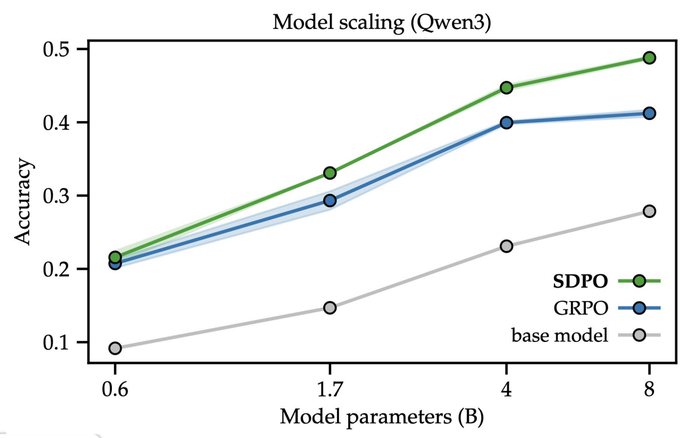
```
Don't include n.
```

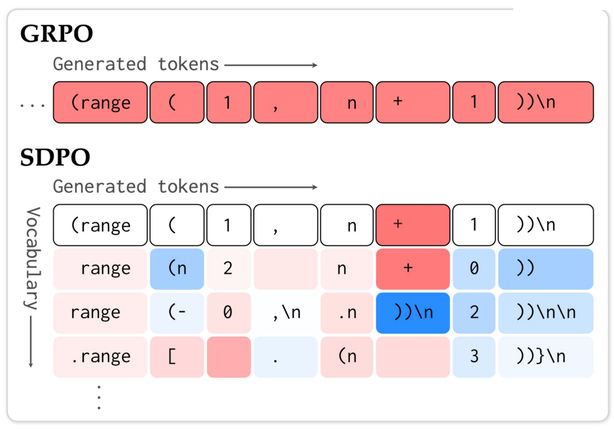
4. Credit assignment by **self-teacher** $\pi_\theta(y \mid x, f)$

Results with Rich Feedback

Setting: Qwen3-8B on LiveCodeBench-v6.

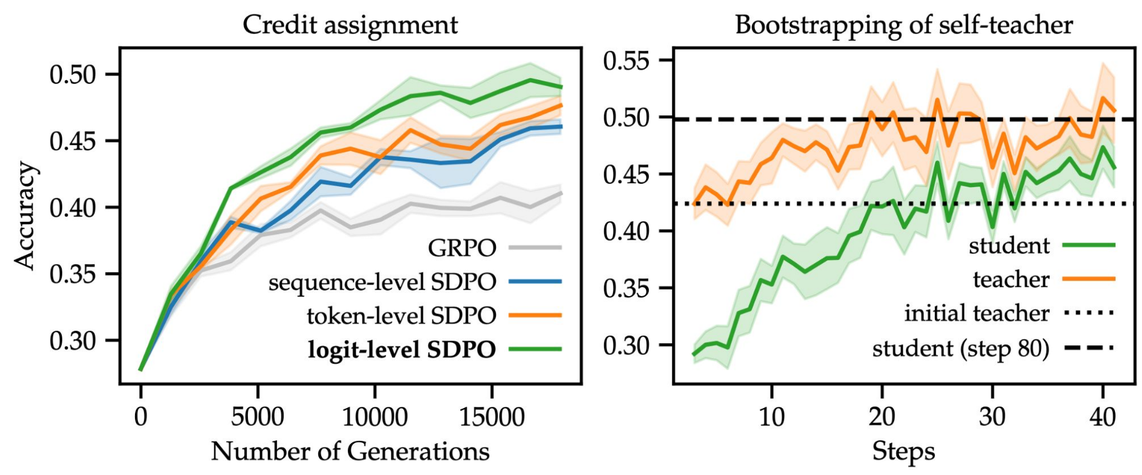
Takeaway: SDPO scales with stronger in-context learners.





Two differences of SDPO to GRPO:

1. Receiving a rich feedback signal.
2. Turning this into dense supervision.

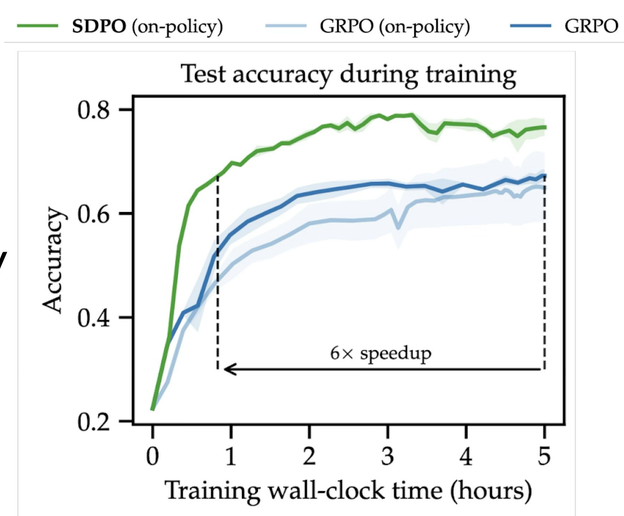


Takeaways: (1) rich feedback and dense credit assignment are complementary, (2) the teacher improves during training.

Results in the RLVR setting

Self-distillation is useful without rich feedback too.

Idea: The teacher reflects on wrong attempts, e.g., by comparing to correct attempts from the same rollout group.



	Chemistry		Physics		Biology		Materials		Tool use	
	1h	5h	1h	5h	1h	5h	1h	5h	1h	5h
Qwen3-8B	41.2		59.2		30.8		58.9		57.5	
+ GRPO	65.9	74.5	63.8	72.7	35.1	**59.9**	**74.3**	77.1	64.9	67.7
+ GRPO (on-policy)	63.3	63.4	63.6	63.6	49.8	49.8	73.9	74.1	60.2	65.7
+ **SDPO** (on-policy)	**73.2**	**80.9**	**66.6**	**75.6**	**50.6**	56.8	72.1	**78.4**	**68.0**	**68.5**
Olmo3-7B-Instruct	22.8		37.7		16.2		36.7		39.3	
+ GRPO	39.7	56.7	55.3	63.3	35.6	**55.8**	70.9	75.0	56.4	**65.0**
+ GRPO (on-policy)	51.4	57.5	**62.7**	62.7	**49.8**	49.8	73.3	73.5	56.8	60.6
+ **SDPO** (on-policy)	**68.0**	**80.0**	59.9	**66.1**	48.0	52.8	**73.7**	**79.1**	**60.8**	62.1

Ablation: Which feedback is most informative?

	Teacher before training		Student trained with SDPO	
	↑ Acc. (%)	↓ Same output (%)	↑ Acc. (%)	Avg. entropy
f = output	32.5 ± 0.5	13.7 ± 0.6	39.9 ± 1.1	0.40 ± 0.0
f = own solution	**42.4** ± 1.0	12.1 ± 0.7	42.6 ± 1.3	0.41 ± 0.0
f = output + own solution	**42.5** ± 1.2	**10.1** ± 0.2	**48.3** ± 1.4	0.38 ± 0.0
f = y + output + own solution	39.3 ± 0.8	30.0 ± 0.9	44.5 ± 1.3	*0.23* ± 0.0

Test-Time Self-Distillation

Jonas Hübotter[1], Frederike Lübeck*,[1,2], Lejs Behric*,[1], Anton Baumann*,[1], Marco Bagatella[1,2], Daniel Marta[1], Ido Hakimi[1], Idan Shenfeld[3], Thomas Kleine Buening[1], Carlos Guestrin[4], Andreas Krause[1]

[1] ETH zürich [2] MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS [3] MIT [4] Stanford University — Learning & Adaptive Systems

How can we learn to solve hard problems?

Use in-context learning to turn the model into its own teacher, then distill it.

Motivation

Question: How can we learn to solve very hard problems at test-time?

Goal: Accelerate time-to-discovery. The *discovery time* is the number of trials until a solution is found. We want to increase the probability of discovery:

$$\text{discovery@}k := \mathbb{P}(\text{discovery time} \leq k)$$

How can we learn before a solution is found?

- Rich feedback such as runtime errors or failed unit tests indicate *why* an attempt failed.

```
Runtime Error
ZeroDivisionError: division by zero
Line 73 in separateSquares (Solution.py)

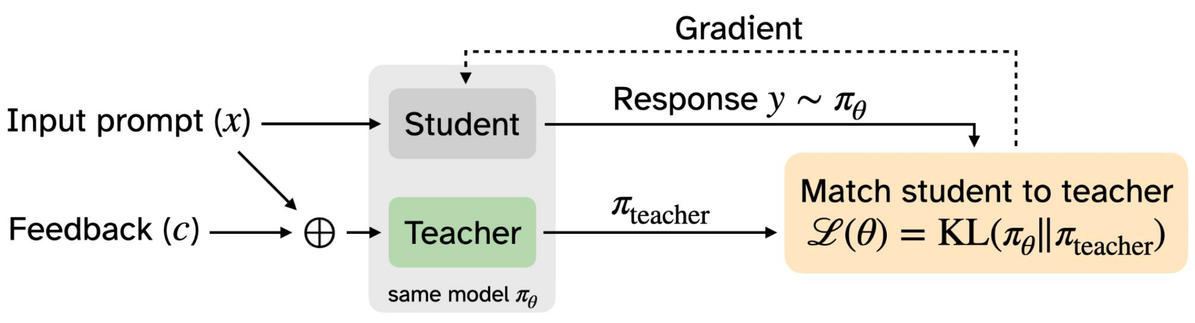
Last Executed Input
[[26,30,2],[11,23,1]]
```

Why current methods fail:

- Information bottleneck of RLVR: Rewards are zero until the solution is found. RLVR methods like GRPO ignore any non-reward feedback from the environment → reducing to best-of-k sampling.
- In-context learning is transient: Models learn from in-context feedback, but performance drops as the context length grows.

Self-Distillation (**SDPO**)

"Leverage in-context learning for in-weight learning."



Two equivalent perspectives:

1. On-policy RL with teacher-based advantages

$$\text{Maximize} \quad A_i(x, y, c) := \log \frac{\pi_\theta(y_i \mid x, c, y_{<i})}{\pi_\theta(y_i \mid x, y_{<i})}$$

2. On-policy distillation with a context-based teacher

$$\text{Minimize} \quad \mathscr{L}_{\text{SDPO}}(\theta) := \frac{1}{|y|} \sum_{i=1}^{|y|} \text{KL}(\pi_\theta(\cdot \mid x, y_{<i}) \| \pi_\theta(\cdot \mid x, c, y_{<i}))$$

Both perspectives yield same gradient in expectation!

Example of Self-Distillation

An example with Qwen3-8B:

1. Question x

```
Write a python function that returns
all numbers from 1 to n. Answer
briefly.
```

2. Answer $y \sim \pi_\theta(\cdot \mid x)$

```
```python
def numbers_up_to_n(n):
    return list(range(1, n + 1))
```
```

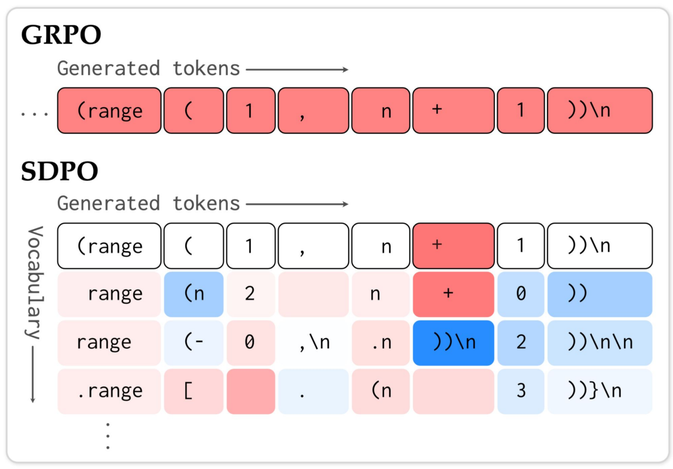
3. Feedback f

```
Don't include n.
```

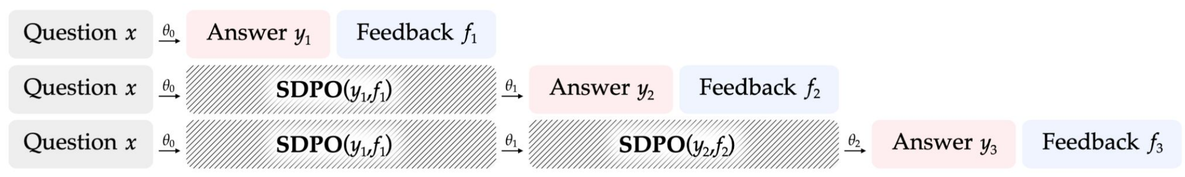
4. Credit assignment by **self-teacher** $\pi_\theta(y \mid x, f)$

Two differences of SDPO to GRPO:

1. Receiving a rich feedback signal.
2. Turning this into dense supervision.



Test-time setting:



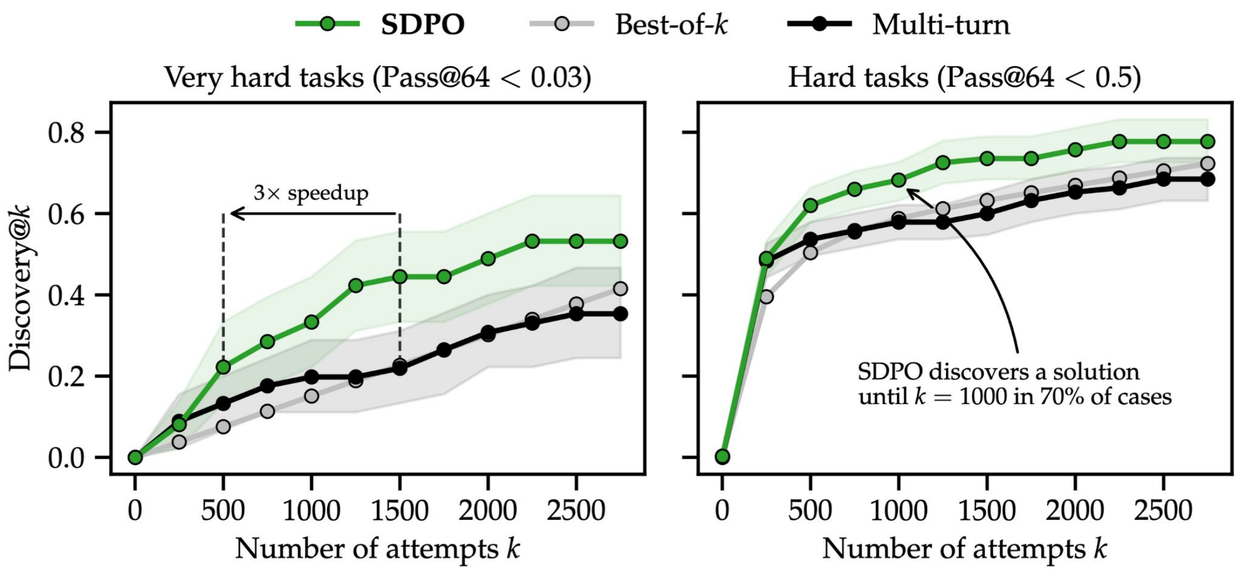
Results

We evaluate Qwen3-8B on hard LiveCodeBench-v6 problems. We define two groups

- Hard tasks with *pass@64 < 0.5*,
- Very hard tasks with *pass@64 < 0.03*.

Baselines:

- Best-of-*k* / GRPO: Sampling *k* times from base.
- Multi-turn: Keeping feedback in context.



Takeaways:

- SDPO significantly increases discovery@k: SDPO solves problems not solvable by the base model.
- The initial teacher does not yet solve questions: Teacher feedback is directionally helpful.
- Multi-turn baseline runs out of context (40k) at *k=800* to *k=1000*. Multi-turn underperforms even prior to running out of context.

Note: pass@k = discovery@k for best-of-k sampling.