A Predictive Law for On-Policy Self-Distillation From World Feedback

Tommy He [1] **Jerome Sieber** [*1] **Matteo Saponati** [*1]

Abstract

Moving beyond simple scalar rewards toward richer world feedback is a natural path to more scalable RL post-training. On-policy self-distillation (OPSD) is a promising recent approach that uses arbitrary feedback as learning signal, yet its reliability compared to established methods, such as GRPO, remains unclear. We identify a strikingly consistent linear correlation between the initial student–self-teacher performance gap and the final performance improvement in OPSD. This relationship holds across context types and model families, providing a powerful predictive law for anticipating the outcome of an OPSD configuration without running the full training procedure. Interestingly, we show that this linear predictability holds with model scale, suggesting a potential basis for new empirical scaling laws on larger models with stronger in-context learning capabilities. In essence, our findings show that OPSD performance can be predicted and tuned before training, offering a principled way to incorporate world feedback as a first-class component of the post-training pipeline.

1. Introduction

Reinforcement learning with verifiable rewards (RLVR) has become a fundamental component of the training pipeline for large language models. Unlike supervised fine-tuning (SFT), RLVR allows the model to sample rollouts, explore on-policy behavior, and train from resulting feedback, leading to enhanced reasoning capabilities (Jaech et al., 2024; Guo et al., 2025; Kimi Team et al., 2025). Despite its success, RLVR is constrained by outcome-level rewards, which are often binary. This limits the use of richer world feedback, such as interpreter messages in coding environments or detailed feedback from environment interactions.

[1]Tufa Labs, Zürich, Switzerland. Correspondence to: Tommy He <tommy@tufalabs.ai>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

To address this limitation, several OPSD approaches have been proposed (Hübotter et al., 2026; Shenfeld et al., 2026; Zhao et al., 2026; Ye et al., 2026a; Sang et al., 2026; Ye et al., 2026b). Rather than relying on scalar rewards, these methods generate dense, logit-level credit assignment directly from arbitrary tokenized feedback, such as the environment and/or the model's own behavior. Despite promising results, OPSD comes with notable challenges: privileged context conditioning shifts the self-teacher's distribution in unpredictable ways, and careful stabilization and context design are required for the approach to work reliably (Zhao et al., 2026; Partridge, 2026; Brown, 2026). For OPSD to become a reliable post-training method, a principled methodology is needed to predict the effect of the variables driving its success. In this work, we provide one such methodology by investigating an important yet overlooked property of OPSD: the privileged context construction. Like standard on-policy distillation (OPD) (Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025), OPSD targets the teacher distribution, but unlike OPD, OPSD allows the teacher's effective performance ceiling to be varied through the choice of the privileged context.

Crucially, because the teacher is the student with privileged context (the *self-teacher*), the initial performance gap between the student and self-teacher (the *student–self-teacher gap*) is cheap to measure before any training. We show empirically that this initial gap is a strong linear predictor of final performance improvement, providing a practical lever to (1) screen multiple privileged context constructions in advance, (2) understand training limits to avoid wasted post-training runs, and most importantly, (3) enable reliable use of world feedback during post-training.

To do so, we focus on coding as a canonical environment with rich feedback (e.g., runtime errors, failed unit tests), using LiveCodeBench as our benchmark (Jain et al., 2024). We post-train open-source models following standard OPSD techniques and investigate the student–self-teacher predictive law through an extensive set of experiments. In sum, our work makes the following contributions:

1. We identify a strong predictive law between the initial student–self-teacher gap and final student performance improvement, enabling early performance estimation without costly full training runs.

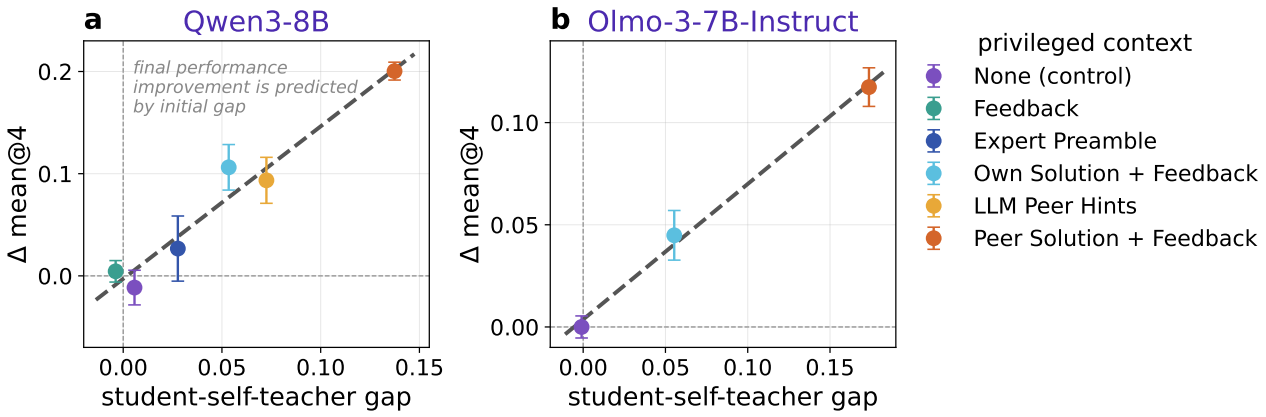

Figure 1. **Linear correlation between initial student–self-teacher gap and final student performance improvement**. **a)** Final student improvement versus the initial gap between the student and its self-generated teacher, for Qwen3-8B. Each dot represents the mean improvement for a given context configuration (see legend and Section 5), with error bars indicating one standard deviation across 3 random seeds. The black dashed line shows the ordinary least squares (OLS) linear fit (intercept $= -0.003$, slope $= 1.492$, $R^2 = 0.949$). The estimated Pearson and Spearman correlations between final student improvement and the initial gap are 0.974 ($p = 0.001$) and 0.886 ($p = 0.019$), respectively. **b)** Same as **a)**, for Olmo-3-7B-Instruct. OLS linear fit: intercept $= 0.004$, slope $= 0.663$, $R^2 = 0.996$. Estimated Pearson and Spearman correlations: 0.998 ($p = 0.043$) and 1.0 ($p = 0.0$), respectively.

2. We demonstrate that this relationship generalizes across different types of privileged information and model families.
3. We show that the relationship holds with model size.

Together, these findings offer a lightweight method for estimating the effectiveness of OPSD configurations during post-training, elevating world feedback as a reliable component of LLM post-training.

2. Preliminaries

Problem formulation. We adopt the OPSD formulation of Hübotter et al. (2026). The student policy π_θ samples rollouts $y \sim \pi_\theta(\cdot \mid x)$ for prompts x drawn from a prompt distribution $\mathcal{D}$. The self-teacher $\pi_{\bar{\theta}}(\cdot \mid x, c)$ shares its weights with the student via an exponential moving average (EMA), where $\bar{\theta}$ slowly drifts toward student parameters θ during training, and is conditioned on privileged context c that contains information unavailable to the student (e.g., world feedback, reference snippets). Although the self-teacher and student have (almost) the same weights, c shifts the teacher's predictive distribution toward stronger behavior through in-context learning, providing dense supervision without the need for a stronger base model. The OPSD objective is the per-token reverse KL divergence between the student and self-teacher under student rollouts:

$$\mathcal{L}_{\text{OPSD}}(\theta; c) = \mathbb{E}_{\substack{y \sim \pi_\theta(\cdot \mid x) \\ x \sim \mathcal{D}}} \left[\frac{1}{|y|} \sum_t \mathrm{KL}\Big(\pi_\theta(\cdot \mid y_{<t}, x) \,\Big\|\, \mathrm{stopgrad}[\pi_{\bar{\theta}}(\cdot \mid y_{<t}, x, c)] \Big) \right], \tag{1}$$

where $\mathrm{stopgrad}$ blocks the gradients through the teacher.

3. Experiments

In our experiments, we perform a series of post-training experiments with OPSD on two model families: Qwen3 (Yang et al., 2025) and Olmo 3 (Olmo et al., 2026). We investigate the design of privileged context c and the resulting performance correlation between the student and self-teacher models. For each model, we start from a pre-trained checkpoint and post-train using OPSD for 50 steps on LiveCodeBench, varying the context c across runs. We construct six distinct contexts c by combining different information sources: the original prompt, the model's own rollouts, peer rollouts from the group, the rollout's success verdict, and external world feedback. The full set of context types is described in Table 1. For each context, we report two metrics: (1) initial student–self-teacher gap: the difference between the self-teacher and student accuracy on the validation set before training, using train-time decoding parameters; and (2) final student improvement (Δ mean@4): the gain in student accuracy on the validation set between start and end of training, using validation-time decoding parameters. More details are provided in Section 5.

3.1. Initial student–self-teacher gap predicts final student accuracy improvements.

We first test how the initial gap between the student and the self-teacher correlates with the student's final performance across different types of privileged contextual information. We use Qwen3-8B and Olmo-3-7B-Instruct as starting checkpoints. We observe a striking correlation between the initial student–self-teacher gap and final performance improvement for both model families (Figure 1). Notably,

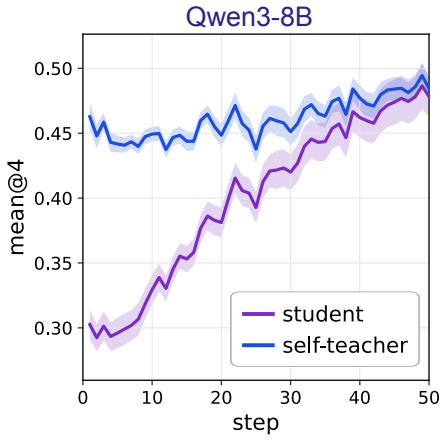

Figure 2. **The student performance gradually approaches the self-teacher performance throughout training.** During post-training of Qwen3-8B on LiveCodeBench with the `Peer Solution + Feedback` privileged context, the student validation accuracy (purple) steadily converges to the self-teacher level (blue). We report the mean (solid line) and standard deviation (shaded region) over 3 seeds.

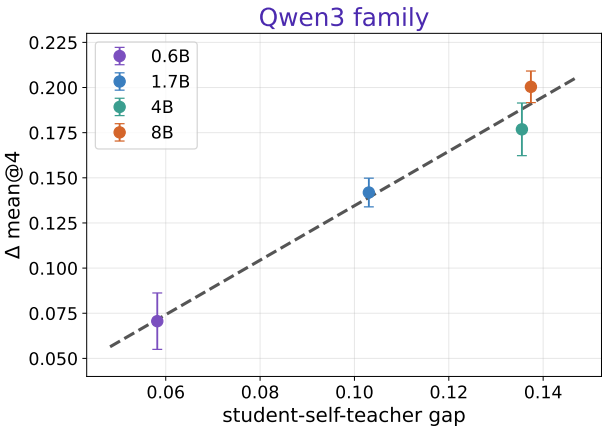

Figure 3. **The predictive law is preserved at scale**. Final student performance improvement vs. initial student–self-teacher gap across model sizes in the Qwen3 family. The initial gap and final performance improvement are computed as in Figure 1. Each dot represents the mean final performance for a given model size, with error bars showing standard deviation over 3 seeds. OLS linear fit: intercept $= -0.016$, slope $= 1.508$, $R^2 = 0.977$ with estimated Pearson and Spearman correlations: 0.988 ($p = 0.012$) and 1.0 ($p = 0.0$).

this linear relationship holds consistently across all tested types of privileged context, despite the qualitatively different content they convey (e.g., environment feedback, peer solutions, or general advice from the self-teacher). Furthermore, we examine the training dynamics and find that the student's evaluation performance steadily converges to the self-teacher's level over time (Figure 2). We also note that we see a rise in self-teacher capabilities over time similar to Hübotter et al. (2026). Finally, we conduct a leave-one-out cross-validation analysis to test the predictive capability of this empirical rule. The rule generalizes well across all folds, with $R^2 = 0.949$ and a root mean squared error (RMSE) of 0.016 for Qwen3-8B and $R^2 = 0.996$ and RMSE $= 0.003$ for Olmo-3-7B-Instruct. This supports a strong and general underlying connection between the student–self-teacher gap and final student performance improvement.

Takeaway 1

The initial student–self-teacher gap strongly and linearly predicts final model performance improvement across privileged context types and model families.

3.2. The predictive law holds with model size.

Next, we examine whether this predictive law holds across model scale. We use Qwen3 models of 4 sizes (0.6B, 1.7B, 4B, 8B) and measure the initial student–self-teacher gap and final performance improvement for a fixed privileged context type `Peer Solution + Feedback`. The linear relationship between the student–self-teacher gap and final student performance improvement holds consistently across all model scales (Figure 3), suggesting it is a reliable, scale-invariant predictor of final performance gain. Crucially, this not only means that final student performance can be estimated early in training, but also that performance gains at larger model sizes can be accurately predicted using this linear relationship.

Takeaway 2

The student–self-teacher gap is a scale-invariant, linear predictor of final performance improvement, enabling accurate prediction of performance gain across model sizes.

4. Discussion

We introduce a predictive law for estimating final student performance from the initial student–self-teacher gap in OPSD. Empirical validation across Qwen3 and Olmo 3 model families on LiveCodeBench reveals a robust linear correlation between the initial gap and final performance improvement, consistent across model families and scales. This relationship has two practical implications. First, it enables screening multiple privileged information configurations without costly post-training runs. Second, it reliably predicts the benefit of incorporating world feedback during post-training, which is especially valuable in rich environment interaction settings. Together, these allow practitioners to estimate the performance ceiling for different privileged contexts prior to any training.

Furthermore, OPSD may serve as a practical alternative to OPD in settings where useful world feedback is available to bootstrap learning. OPD performance is often non-monotonic as stronger teachers do not always improve the student: reasoning trajectories can diverge (Li et al., 2026), and reverse-KL objectives may destabilize high-capacity students (Ko et al., 2026). In practice, recipes restrict teacher selection, e.g., to matched-size teachers or the same model family (Lu & Thinking Machines Lab, 2025; Ko et al., 2026). As a result, no predictive rules exist for OPD that forecast final performance without running experiments (Song & Zheng, 2026). In contrast, OPSD leverages the intrinsic consistency between student and self-teacher by construction, which we hypothesize explains our clean predictive law.

Our work opens several avenues for future research. First, it would be valuable to further study how different forms of privileged information affect final student performance and where they fall along the empirical predictive relationship. Methods that improve performance via in-context learning are a promising direction, including few-shot learning (Brown et al., 2020), retrieval-augmented generation (Lewis et al., 2021), prompt evolution (Agrawal et al., 2026; Khattab et al., 2023), and context evolution (Zhang et al., 2026). Second, scaling the law to larger models is an important next step to assess its generality. Finally, comparing the predicted gains from OPSD across different and richer environments would further establish its practical utility.

5. Methodology

5.1. Self-Teacher Construction Configurations

We study six privileged contexts c that span qualitatively distinct information sources (Table 1). Notably, the configurations differ in *kind*, not merely in strength, which allows the linear correlation in Figure 1 to test whether the scalar gap alone is a sufficient statistic for final performance regardless of the mechanism that produced it.

Table 1. The six self-teacher constructions studied. Additional details in the Supplementary Materials (Table S2).

Self-Teacher Construction
`None (control)`
`Expert Preamble`
`Feedback`
`LLM Peer Hints`
`Own Solution + Feedback`
`Peer Solution + Feedback`

5.2. Models, Training, and Evaluation

We use Qwen3-{0.6B, 1.7B, 4B, 8B} (Yang et al., 2025) in non-thinking mode and Olmo-3-7B-Instruct (Olmo et al., 2026) as base policies. Each model is post-trained with OPSD for 50 gradient steps, batch size 32, and 8 rollouts per prompt; full hyperparameters are listed in Appendix A. We evaluate on LiveCodeBench v6 (Jain et al., 2024) (problems from February–May 2025) using mean@4 pass rate.

6. Related Work

RL with LLMs. Large-scale RL has significantly improved LLM performance on general reasoning tasks via RLVR methods such as GRPO (Shao et al., 2024). A complementary line of research seeks finer-grained credit assignment by training dedicated process reward models (PRMs) that score individual reasoning steps rather than final answers (Setlur et al., 2025). While RLVR approaches rely on outcome-level feedback and PRMs introduce significant overhead from a separate model, OPSD sidesteps both limitations by using the model as its own supervisor, requiring no external reward model.

On-policy distillation. When a capable teacher model is available, distillation offers a compelling alternative to standard SFT (Hinton et al., 2015). OPD represents a strong implementation of distillation because it avoids the exposure-bias gap of evaluating the student under the teacher's state distribution, and it is cheaper to sample from (Yang et al., 2025; Lu & Thinking Machines Lab, 2025). By construction, OPD is bounded by the teacher distribution in the limit. By contrast, OPSD is bounded by (1) the performance gain from additional context, which can be adjusted dynamically during training, and (2) the self-teacher's capabilities, which can improve over time and raise the ceiling accordingly.

On-policy self-distillation. Our work is fundamentally related to OPSD approaches. In SDFT (Shenfeld et al., 2026), the self-teacher is constructed with expert demonstrations, allowing the model to distill task-relevant knowledge from in-context examples into its base parameters. In RLTF-SD (Song et al., 2026), the authors use textual feedback from annotators or users for the self-teacher, enabling the model to internalize corrective critiques. In ERL (Shi et al., 2026), the self-teacher is constructed with a structured reflection derived from environmental feedback on a failed first attempt, guiding a refined second attempt whose success is then consolidated into the policy. In OPCD (Ye et al., 2026b), historical solution traces or optimized system prompts are the self-teacher's privileged context, training the student to internalize experiential knowledge. In OPSD (Zhao et al., 2026), the self-teacher is constructed with verified correct reasoning traces. Finally, in SDPO (Hübotter et al., 2026), the self-teacher is conditioned on rich textual feedback from verifiable environments and correct model behavior. While these methods differ in their context configurations, they all share the same core approach: distilling in-context learning capabilities with privileged context into the student model.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Agarwal, R., Vieillard, N., Zhou, Y., Stanczyk, P., Garea, S. R., Geist, M., and Bachem, O. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

Agrawal, L. A., Tan, S., Soylu, D., Ziems, N., Khare, R., Opsahl-Ong, K., Singhvi, A., Shandilya, H., Ryan, M. J., Jiang, M., Potts, C., Sen, K., Dimakis, A. G., Stoica, I., Klein, D., Zaharia, M., and Khattab, O. Gepa: Reflective prompt evolution can outperform reinforcement learning, 2026. URL https://arxiv.org/abs/2507.19457.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Brown, W. On sft, rl, and on-policy distillation. Post on X (formerly Twitter), April 2026. URL https://x.com/willccbb/status/2050038277454143918.

Guo, D., Yang, D., Zhang, H., Song, J., et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hübotter, J., Lübeck, F., Behric, L., Baumann, A., Bagatella, M., Marta, D., Hakimi, I., Shenfeld, I., Buening, T. K., Guestrin, C., et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., et al. OpenAI o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

Jain, N., Han, K., Gu, A., Li, W.-D., Yan, F., Zhang, T., Wang, S., Solar-Lezama, A., Sen, K., and Stoica, I. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Khattab, O., Singhvi, A., Maheshwari, P., Zhang, Z., Santhanam, K., Vardhamanan, S., Haq, S., Sharma, A., Joshi, T. T., Moazam, H., Miller, H., Zaharia, M., and Potts, C. Dspy: Compiling declarative language model calls into self-improving pipelines, 2023. URL https://arxiv.org/abs/2310.03714.

Kimi Team, Du, A., Gao, B., Xing, B., Jiang, C., et al. Kimi k1.5: Scaling reinforcement learning with LLMs, 2025. URL https://arxiv.org/abs/2501.12599.

Ko, J., Abdali, S., Kim, Y. J., Chen, T., and Cameron, P. Scaling reasoning efficiently via relaxed on-policy distillation, 2026. URL https://arxiv.org/abs/2603.11137.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. URL https://arxiv.org/abs/2005.11401.

Li, Y., Zuo, Y., He, B., Zhang, J., Xiao, C., Qian, C., Yu, T., ang Gao, H., Yang, W., Liu, Z., and Ding, N. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe, 2026. URL https://arxiv.org/abs/2604.13016.

Lu, K. and Thinking Machines Lab. On-policy distillation. Thinking Machines Lab Blog (Connectionism), October 2025. URL https://thinkingmachines.ai/blog/on-policy-distillation. Blog post.

Olmo, T., :, Ettinger, A., Bertsch, A., Kuehl, B., Graham, D., Heineman, D., Groeneveld, D., Brahman, F., Timbers, F., Ivison, H., Morrison, J., Poznanski, J., Lo, K., Soldaini, L., Jordan, M., Chen, M., Noukhovitch, M., Lambert, N., Walsh, P., Dasigi, P., Berry, R., Malik, S., Shah, S., Geng, S., Arora, S., Gupta, S., Anderson, T., Xiao, T., Murray, T., Romero, T., Graf, V., Asai, A., Bhagia, A., Wettig, A., Liu, A., Rangapur, A., Anastasiades, C., Huang, C., Schwenk, D., Trivedi, H., Magnusson, I., Lochner, J., Liu, J., Miranda, L. J. V., Sap, M., Morgan, M., Schmitz, M., Guerquin, M., Wilson, M., Huff, R., Bras, R. L., Xin, R., Shao, R., Skjonsberg, S., Shen, S. Z., Li, S. S., Wilde, T., Pyatkin, V., Merrill, W., Chang, Y., Gu, Y., Zeng, Z., Sabharwal, A., Zettlemoyer, L., Koh, P. W., Farhadi, A., Smith, N. A., and Hajishirzi, H. Olmo 3, 2026. URL https://arxiv.org/abs/2512.13961.

Partridge, H. Opsd isn't a silver bullet for continual learning. Post on X (formerly Twitter), March 2026.

URL https://x.com/part_harry_/status/2038715548071325794.

Sang, H., Xu, Y., Zhou, Z., He, R., Wang, Z., and Sun, J. On-policy self-distillation for reasoning compression. *arXiv e-prints*, pp. arXiv–2603, 2026.

Setlur, A., Rajaraman, N., Levine, S., and Kumar, A. Scaling test-time compute without verification or rl is suboptimal. *arXiv preprint arXiv:2502.12118*, 2025.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Shenfeld, I., Damani, M., Hübotter, J., and Agrawal, P. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297. ACM, March 2025. doi: 10.1145/3689031.3696075. URL http://dx.doi.org/10.1145/3689031.3696075.

Shi, T., Chen, S., Jiang, B., Song, L., Yang, L., and Zhao, J. Experiential reinforcement learning. *arXiv preprint arXiv:2602.13949*, 2026.

Song, M. and Zheng, M. A survey of on-policy distillation for large language models. *arXiv preprint arXiv:2604.00626*, 2026.

Song, Y., Chen, L., Tajwar, F., Munos, R., Pathak, D., Bagnell, J. A., Singh, A., and Zanette, A. Expanding the capabilities of reinforcement learning via text feedback. *arXiv preprint arXiv:2602.02482*, 2026.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Ye, T., Dong, L., Dong, Q., Wu, X., Huang, S., and Wei, F. Online experiential learning for language models. *arXiv preprint arXiv:2603.16856*, 2026a.

Ye, T., Dong, L., Wu, X., Huang, S., and Wei, F. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 2026b.

Zhang, Q., Hu, C., Upasani, S., Ma, B., Hong, F., Kamanuru, V., Rainton, J., Wu, C., Ji, M., Li, H., Thakker, U., Zou, J., and Olukotun, K. Agentic context engineering: Evolving contexts for self-improving language models, 2026. URL https://arxiv.org/abs/2510.04618.

Zhao, S., Xie, Z., Liu, M., Huang, J., Pang, G., Chen, F., and Grover, A. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

A. Experimental Details

The experiments provided in Section 3 are all performed on LiveCodeBench v6 (Jain et al., 2024) (problems from February–May 2025). We build on the SDPO[1] (Hübotter et al., 2026) and VeRL[2] (Sheng et al., 2025) codebases. We use the same set of hyperparameters for all our experiments, which are provided in Table S1.

Table S1. Hyperparameters.

Parameter	**Value**
Model and data	
Base models	Qwen3, Olmo3
Dataset	LiveCodeBench v6
Training	
Train batch size (prompts)	32
Mini-batch size	1
Optimizer	AdamW
Learning rate	1×10^{-6}
LR schedule	constant
Weight decay	0.01
Adam (β_1, β_2)	$(0.9,\ 0.999)$
Gradient clip (max norm)	1.0
Sampling — training rollouts	
Rollouts per prompt (n)	8
Temperature	1.0
Top-p	1.0
Max prompt length	2048
Max response length	32,768
Thinking mode	disabled
Sampling — validation	
Samples per prompt (n)	4
Temperature	0.6
Top-p	0.95
Max prompt length	2048
Max response length	32,768
Thinking mode	disabled
SDPO	
Divergence	Reverse-KL
Teacher EMA rate	0.01
Distillation top-k	20

Below, we provide more information on the specific experiments we ran to produce the figures in the main text.

1. For the experiments in Subsection 3.1, we run OPSD for 50 steps on Qwen3-8B for 3 seeds and Olmo-3-7B-Instruct for 2 seeds while varying the context according to the self-teacher constructions in Table S2. We ran additional experiments with more context constructions, where the same law holds well, but omit them because they lack sufficient seed coverage.

2. For the experiments in Subsection 3.2, we run OPSD for 50 steps on Qwen3-{0.6B, 1.7B, 4B, 8B} for 3 seeds while fixing the privileged context type `Peer Solution + Feedback`. We note that we did initial experiments on Qwen3-14B; however, we believe that for larger models we might have to tune hyperparameters separately and hence do not include these inconclusive results here.

[1] https://github.com/lasgroup/SDPO
[2] https://github.com/verl-project/verl

Table S2. Six self-teacher constructions: {prompt} is the original problem; {sol} is a correct solution (peer's, the rollout's own, or LLM-extracted hints, depending on the row); {fb} is environment feedback. Bracketed sections are conditional per sample.

Self-Teacher Construction	Template
None (control)	{prompt}
Expert Preamble	[expert preamble: DS/algo checklist; complexity-vs-input-size table; pattern rules; edge cases; 9-step methodology] --- {prompt} Think step by step first, then write the solution.
Feedback	{prompt} The following is feedback from your unsuccessful earlier attempt: {fb} Correctly solve the original question.
LLM Peer Hints	{prompt} Correct solution: {sol} Correctly solve the original question.
Own Solution + Feedback	{prompt} [Correct solution: {sol}] (if rollout correct) [Feedback: {fb}] (else) Correctly solve the original question.
Peer Solution + Feedback	{prompt} Correct solution: {sol} The following is feedback from your unsuccessful earlier attempt: {fb} Correctly solve the original question.

Computational Resources All experiments ran on a cluster with NVIDIA B200 GPUs. We estimate each run took around 20h, leading to a total runtime of approximately 720 B200-hours, plus roughly 240 B200-hours for debugging and failed runs.