ON-POLICY SELF-DISTILLATION VIA ACCUMULATED EXPERIENCE REPRESENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-Policy Self-Distillation (OPSD) has achieved strong performance in language-model post-training by distilling reference-conditioned teacher predictions along student-generated trajectories. However, existing methods still typically rely on explicit references or solutions to construct supervision, leaving open how to further improve self-distillation without introducing such additional supervision. In this work, we investigate whether a model can use its own past responses as experience and turn them into self-distillation supervision. We find that a single self-generated experience can substantially alter the current prediction, but this effect is sensitive to which experience is sampled; aggregating multiple experiences markedly reduces this prediction variation. Based on this observation, we propose **Accumulated Experience Self-Distillation (AESD)**. AESD aggregates predictions induced by multiple past experiences and progressively accumulates their predictive effects throughout training through a shared set of continuously updated Soft tokens, forming a reusable distillation target. The resulting target is then distilled into the student along its own on-policy trajectories, while neither the experience buffer nor the Soft tokens are required at inference time. Across seven recent mathematical reasoning benchmarks, AESD outperforms competitive baselines on both Qwen3-4B and Qwen3-8B, improving average accuracy by 1.34 and 0.69 percentage points, while also increasing average APT by 2.43% and 2.06%, respectively.

1 INTRODUCTION

Recently, On-Policy Distillation (OPD) (Agarwal et al., 2024; Gu et al., 2024; Lu & Lab, 2025; Li et al., 2026a) has attracted increasing attention as an important paradigm for post-training large language models. Unlike conventional distillation on static teacher-generated data, OPD distills teacher supervision directly on student-generated on-policy trajectories. To further improve training efficiency, recent work extends OPD to On-Policy Self-Distillation (OPSD) (Zhao et al., 2026; Ye et al., 2026b), which uses a privileged teacher policy from the same model to provide dense on-policy supervision without requiring a separate large teacher. As illustrated in Fig. 1(a), with access to the GT reference, the teacher produces next-token predictions that are then distilled into the student. This self-distillation paradigm has since been extended beyond mathematical reasoning to generative modeling (Jiang et al., 2026), agentic tool use (Wang et al., 2026a), multimodal reasoning (Yuan et al., 2026), and robotic manipulation (Wang et al., 2026b).

Despite its effectiveness, directly matching the teacher prediction conditioned on the reference can produce highly concentrated supervision. In this setting, a small number of tokens with large gaps between teacher and student predictions dominate the distillation objective, which motivates the pointwise KL clipping adopted in prior work (Zhao et al., 2026; Liu et al., 2026). More recently, OP^{2}SD (Ichihara et al., 2026) demonstrates that self-distillation remains effective even when the current problem's reference is replaced with a solution from another problem, as illustrated in Fig. 1(b). However, additional information can alter model predictions independently of whether it is useful for the current problem: irrelevant information can substantially degrade reasoning performance (Yue et al., 2025), while even randomly occurring tokens in the input can receive systematically increased prediction probabilities (Niu et al., 2025). Therefore, when the solution comes from another problem, whether the resulting teacher prediction should still serve as a direct learning target for the current student becomes less clear. This motivates a simple question:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

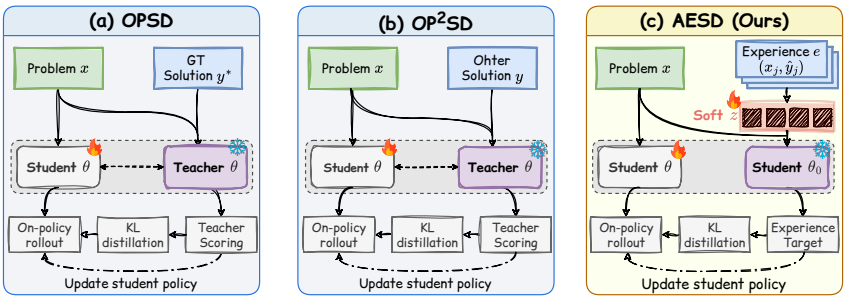

Figure 1: From reference-based self-distillation to accumulated experience. (a) OPSD uses the current problem's GT reference. (b) OP^{2}SD replaces it with a solution from another problem. (c) AESD learns from accumulated self-generated experience.

Can a model learn from how it answered previous problems and reuse that experience for subsequent self-distillation?

To answer this question, we first examine how the model's own past responses affect its current predictions. As shown in Fig. 2, when only a single self-generated experience is used, the induced prediction varies noticeably depending on which experience is sampled. Aggregating more experiences consistently reduces this variation, indicating that the predictive effect of a single past response is strongly sample-dependent and can be stabilized through aggregation. Motivated by this observation, we propose Accumulated Experience Self-Distillation (AESD), which consolidates the predictive effects repeatedly induced by the model's own past responses rather than relying on a single cross-problem solution. As illustrated in Fig. 1(c), AESD treats self-generated responses to previous problems as experience and uses a frozen copy of the initial student to measure how these experiences change the model's prediction at the current student state. Predictions induced by multiple experiences are first aggregated within each state, while a shared set of learnable Soft tokens further accumulates their predictive effects across training. These effects are progressively encoded into a reusable Soft representation, which defines a distillation target for the student along its own on-policy trajectories. At inference time, neither the experience buffer nor the Soft tokens are required, leaving only the updated student. AESD therefore requires neither GT references nor correctness labels.

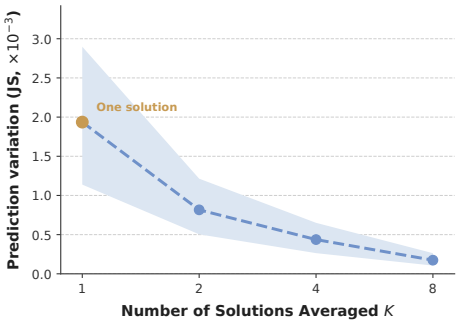

Figure 2: Aggregating multiple experiences reduces prediction variation. Prediction variation decreases as more self-generated experiences are aggregated.

Across seven recent mathematical reasoning benchmarks, AESD consistently improves the overall performance of both Qwen3-4B and Qwen3-8B, achieving average accuracies of 53.89% and 55.94%, respectively, and outperforming both reference-conditioned OPSD and cross-problem OP^{2}SD. It also achieves the highest average Accuracy-per-Token on both model scales.

Our contributions are summarized as follows:

- We investigate whether a model can reuse its own past responses for self-distillation, and find that single self-generated experiences are highly sample-dependent, while aggregating multiple experiences reduces this variation.
- We propose Accumulated Experience Self-Distillation (AESD), which aggregates predictive effects from self-generated experiences through a learnable Soft representation. AESD requires neither GT references nor correctness labels and adds no inference-time input.
- We evaluate AESD on Qwen3-4B and Qwen3-8B across seven recent mathematical reasoning benchmarks. AESD improves average accuracy and token efficiency, while further analyses confirm the benefits of cross-batch accumulation and transfer to unseen problems.

2 Related Work

On-Policy Self-Distillation. On-policy distillation (OPD) (Gu et al., 2024; Agarwal et al., 2024) trains students on their own generated trajectories while relying on a separate teacher for token-level supervision, thereby reducing the mismatch between training and inference distributions. Building on this paradigm, On-Policy Self-Distillation (OPSD) (Zhao et al., 2026) removes the need for an external teacher by using the same model, conditioned on a verified reference solution, as the teacher for self-distillation. Subsequent studies (Shenfeld et al., 2026; Hübotter et al., 2026) extend this idea to other forms of conditioning information, including demonstrations and environmental feedback. More recent work further revisits the necessity of paired reference solutions: U-OPSD (Li et al., 2026b) constructs pseudo-solutions solely from multiple rollouts generated by the model itself, while OP^{2}SD and related studies show that solutions from other problems can also provide effective supervision, suggesting that the gains of OPSD do not fully depend on privileged information strictly paired with the current problem (Ichihara et al., 2026; Shrestha & Tessier, 2026). Along this direction, we move from a single reference context to cross-problem historical experience, aggregate its effects on the predictive distribution through learnable soft representations, and transform the aggregated predictive effects into self-distillation targets.

Context Distillation transfers behaviors induced by additional context into model parameters. Early works distill task instructions, examples, scratchpads, or contextual knowledge into model parameters (Askell et al., 2021; Snell et al., 2022; Caccia et al., 2025; Kujanpää et al., 2025). More recently, On-Policy Context Distillation (OPCD) (Ye et al., 2026a) combines context distillation with on-policy learning by distilling a context-conditioned teacher along student-generated trajectories. For experiential knowledge distillation, it explicitly extracts transferable natural-language experience from historical solution traces and accumulates it as textual context. Our work differs in how historical experience is represented: rather than extracting and concatenating textual experience, we aggregate its effects on predictive distributions through learnable soft representations and distill the resulting calibrated targets.

3 Preliminary

3.1 Background

On-Policy Self-Distillation (OPSD). OPSD provides dense token-level supervision on trajectories generated by the student policy itself. Consider a reasoning dataset $\mathcal{D} = \{(x_i, y_i^\star)\}_{i=1}^N$, where x_i denotes a problem and $y_i^\star$ its reference solution. Given a problem x, the student policy π_θ first generates an on-policy trajectory $\hat{y} \sim \pi_\theta(\cdot \mid x)$. At each token position t, the student and a privileged teacher evaluate the same student-generated prefix $\hat{y}_{<t}$, but under different conditioning contexts:

$$p_t^S = \pi_\theta(\cdot \mid x, \hat{y}_{<t}), \qquad p_t^T = \pi_{\bar{\theta}}(\cdot \mid x, y^\star, \hat{y}_{<t}), \tag{1}$$

where $\pi_{\bar{\theta}}$ is instantiated from the same underlying model but additionally conditions on the privileged reference solution $y^\star$. The teacher distribution is treated as a fixed target during student optimization. OPSD then trains the student to match the teacher distribution along its own generated trajectory:

$$\mathcal{L}_{\text{OPSD}} = \mathbb{E}_{(x,y^\star)\sim\mathcal{D},\, \hat{y}\sim\pi_\theta(\cdot|x)} \left[\frac{1}{|\hat{y}|} \sum_{t=1}^{|\hat{y}|} D\left(p_t^T \parallel p_t^S\right) \right], \tag{2}$$

where $D(\cdot \parallel \cdot)$ denotes a divergence between the full-vocabulary next-token distributions. Unlike conventional knowledge distillation with a separately trained teacher model, OPSD derives its supervision from the same underlying model under privileged conditioning. Since both distributions are evaluated on states visited by the current student, OPSD combines on-policy learning with dense token-level distributional supervision.

4 Method

In this section, we introduce AESD, an accumulated experience self-distillation framework. We first construct an **Experience Distribution** in Sec. 4.1 to measure how historical responses from other

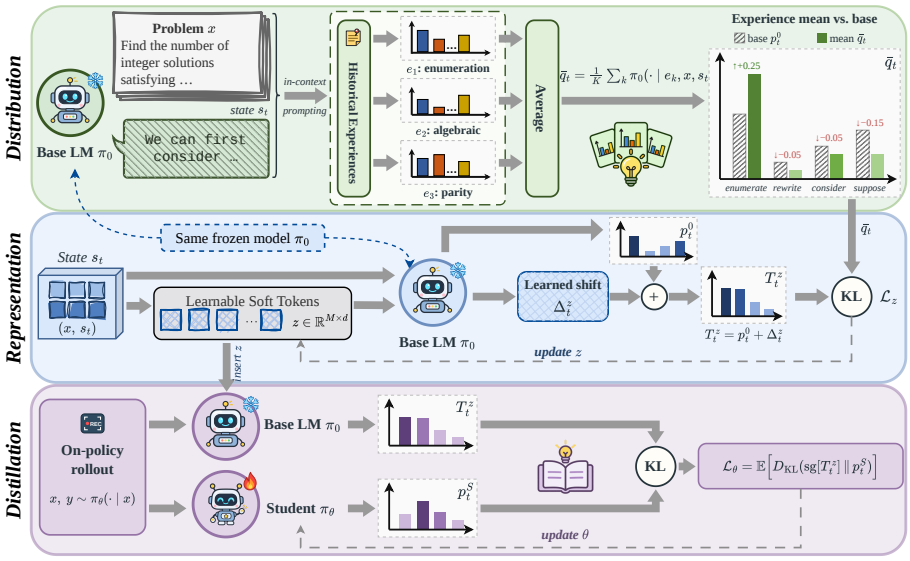

Figure 3: AESD training pipeline. Historical experiences define an aggregated predictive target; shared soft tokens accumulate its predictive effect, and the resulting calibrated target supervises the student.

problems affect the model's prediction at the current student state. We then present **Experience Representation** in Sec. 4.2, which accumulates these predictive changes into learnable soft tokens and converts them into a calibrated target distribution. Finally, Sec. 4.3 introduces **Experience Distillation**, which transfers the resulting target into the student parameters. An overview of AESD is illustrated in Fig. 3.

4.1 Experience Distribution

The core idea of AESD is that the historical responses generated by the model when answering previous problems can provide useful experience for subsequent learning. Instead of directly treating these historical responses as reference answers for the current problem, we focus on how they change the model's predictive distribution at the current state when provided as additional context. Given a current problem x_i, the student policy π_θ first generates an on-policy response

$$\hat{y}_i = (\hat{y}_{i,1}, \ldots, \hat{y}_{i,T}) \sim \pi_\theta(\cdot \mid x_i). \tag{3}$$

At position t, we denote the resulting student-generated state as $s_t = (x_i, \hat{y}_{i,<t})$. A historical experience from another problem is denoted by $e_j = (x_j, \hat{y}_j)$, where $j \neq i$ and $\hat{y}_j$ is the complete response previously generated for problem x_j. Since x_i is fixed, we omit the index i for simplicity. Let π_0 denote a frozen copy of the initial student policy. We first compute the prediction without any historical experience:

$$p_t^0(\cdot) = \pi_0(\cdot \mid s_t). \tag{4}$$

For a historical experience e_j, we provide it as additional context to the same frozen policy and obtain

$$q_{j,t}(\cdot) = \pi_0(\cdot \mid e_j, s_t), \qquad j \neq i. \tag{5}$$

Here, p_t^0 denotes the original prediction, while $q_{j,t}$ denotes the prediction after incorporating the historical experience e_j.

As shown in Fig. 2, predictions conditioned on a single historical response can be sensitive to the particular experience sampled. To reduce this sampling dependence, we sample K historical experiences from other problems, with their index set denoted by $\mathcal{J}_i$, and aggregate their corresponding predictions:

$$\bar{q}_t(\cdot) = \frac{1}{K}\sum_{j\in\mathcal{J}_i} q_{j,t}(\cdot), \qquad |\mathcal{J}_i| = K. \tag{6}$$

We refer to $\bar{q}_t$ as the *experience distribution*, which represents the aggregated predictive effect of multiple historical experiences at the current state. By default, we use $K = 2$. We define the predictive change of the experience distribution relative to the original prediction:

$$\delta_t^{\mathcal{B}} = \log \bar{q}_t - \log p_t^0. \tag{7}$$

4.2 Experience Representation

Since the experience distribution $\bar{q}_t$ varies across states and sampled experiences, AESD introduce learnable soft tokens $z \in \mathbb{R}^{M \times d}$ to accumulate predictive changes throughout training, where M is the number of soft tokens and d is the model embedding dimension. Let z_0 denote the random initialization of the soft tokens. At the current state s_t, we evaluate the frozen initial policy π_0 under the initial and current soft tokens:

$$q_t^{z_0} = \pi_0(\cdot \mid z_0, s_t), \qquad q_t^{z} = \pi_0(\cdot \mid z, s_t). \tag{8}$$

We measure the predictive change as the soft tokens evolve from their initialization z_0 to the current parameters z:

$$\delta_t^z = \log q_t^z - \log q_t^{z_0}. \tag{9}$$

Since δ_t^z captures the change relative to the soft initialization, we apply it to the original prediction p_t^0 to remove the shift introduced by the random initialization and construct the soft target:

$$T_t^z = \mathrm{softmax}\left(\log p_t^0 + \delta_t^z\right). \tag{10}$$

We optimize z using the experience distribution $\bar{q}_t$ from Eq. 6:

$$\mathcal{L}_z = \mathbb{E}_t \left[\underbrace{D_{\mathrm{KL}}\left(\bar{q}_t \| T_t^z\right)}_{\text{experience matching}} + \lambda \underbrace{D_{\mathrm{KL}}\left(T_t^z \| p_t^0\right)}_{\text{base regularization}} \right]. \tag{11}$$

The first term transfers the aggregated experience signal into the soft representation, allowing z to accumulate predictive changes across training states. The second term anchors T_t^z to the original prediction p_t^0, preventing excessive drift. Notably, AESD does not require token-level KL clipping. Experience aggregation and base regularization instead limit the deviation of the training target from the original prediction.

4.3 Experience Distillation

In AESD, the soft tokens z are used only during training. To remove the need for historical experiences or additional soft tokens at inference time, we distill the experience-derived target T_t^z into the student parameters. We refer to this process as *Experience Distillation*. After updating z using Eq. 11, we recompute T_t^z according to Eq. 10. At the same student-generated state s_t, the current student prediction is $p_t^\theta = \pi_\theta(\cdot \mid s_t)$. We then treat T_t^z as a fixed supervision target and optimize

$$\mathcal{L}_\theta = \mathbb{E}_t \left[D_{\mathrm{KL}}\left(\mathrm{sg}[T_t^z] \| p_t^\theta\right) \right]. \tag{12}$$

The student always receives only the original state s_t. Thus, Experience Distillation internalizes the predictive changes accumulated by z into the student parameters without introducing additional inference-time context.

During training, we alternate the updates of z and θ on the same batch of on-policy trajectories. We first update z using Eq. 11, recompute T_t^z, and then update θ using Eq. 12. As the student policy evolves, we periodically regenerate on-policy trajectories. At inference time, both the historical experience buffer and the soft tokens are removed, leaving only the trained student policy π_θ.

5 Experiments

5.1 Experimental Setup

Models and training data. We evaluate AESD on Qwen3-4B and Qwen3-8B (Yang et al., 2025). Following the OPSD setting (Zhao et al., 2026), we use the same training configuration unless noted.

Table 1: Accuracy and token efficiency on Qwen3-4B. We report Avg@12 (%) and APT; higher is better for both metrics. Best results within each benchmark are in bold.

	2025			2026				
Method	ARML	MPfG	Apex	AIME	BrUMO	CMIMC	ArXivMath	Avg.
Avg@12 (%) ↑								
Qwen	**75.42**	72.08	8.33	64.72	64.44	68.40	13.96	52.48
OPSD	72.50	70.83	9.04	65.28	**68.33**	67.71	14.17	52.55
OP^{2}SD	71.67	71.67	9.40	66.11	67.22	64.93	14.17	52.17
AESD	**75.42**	**72.50**	**9.93**	**67.22**	67.78	**68.75**	**15.62**	**53.89**
Accuracy per Token (APT, $\times 10^{-5}$*)* ↑								
Qwen	4.859	4.396	0.382	3.885	4.211	4.662	0.863	3.323
OPSD	4.998	4.535	0.414	3.997	4.469	4.643	0.859	3.416
OP^{2}SD	4.834	4.618	0.430	4.004	4.471	4.326	0.849	3.362
AESD	**5.032**	**4.652**	**0.462**	**4.051**	**4.500**	**4.808**	**0.989**	**3.499**

Our training problems are drawn from OpenThoughts-Math-30K, a mathematical reasoning subset derived from OpenThoughts (Guha et al., 2026), containing 29,434 examples. AESD uses 2,400 distinct training questions (4,800 student rollouts), accessing only question text without reference solutions or correctness labels. For each student trajectory, we sample self-generated responses to other problems as cross-problem experiences.

Evaluation benchmarks and metrics. We evaluate on seven recent mathematical reasoning benchmarks: *AIME 2026, ARML 2025, BrUMO 2026, CMIMC 2026, Math Prize for Girls 2025, ArXivMath May 2026, and Apex-Shortlist 2025* (Dekoninck et al., 2026). We focus on benchmarks from 2025 and 2026 to reduce reliance on long-established test sets that may be more susceptible to benchmark contamination. We report **Avg@12 accuracy** and **Accuracy per Token (APT)**, where APT measures the number of correct predictions normalized by the total number of generated tokens. All methods are evaluated with thinking enabled under identical decoding parameters and generation limits. For the ablation studies in Sec. 5.3, we use a fixed subset consisting of AIME 2026, ARML 2025 (Individual and Team), BrUMO 2026 Div-A, and ArXivMath May 2026, comprising 105 problems in total. Benchmark subsets and metric definitions are detailed in Appendix A.

Baselines. We compare AESD with the original Qwen3 models, OPSD (Zhao et al., 2026), and OP^{2}SD (Ichihara et al., 2026). OPSD conditions its teacher on a reference solution to the current problem, whereas OP^{2}SD replaces the paired reference with a worked solution to a different problem. These comparisons contrast paired reference conditioning, explicit cross-problem conditioning, and AESD's learned representation of self-generated experiences.

Implementation details. AESD aggregates $K = 2$ cross-problem experiences and uses $M = 8$ shared learnable soft tokens. For each training batch, we first update the soft representation and then recompute the distillation target to update the student. We train for 150 student updates with thinking disabled and a maximum rollout length of 1,024 tokens. All trained methods use LoRA with rank 64 and a scaling parameter of 128. We evaluate checkpoints every 25 updates to track test performance over training. The training steps corresponding to the main results are listed in Appendix A. For each method and model size, the same reported checkpoint is evaluated on all seven benchmarks and used for both metrics. Training budgets, optimization settings, and the evaluation schedule are provided in Appendix A.

5.2 Main Results (Does Accumulated Experience Improve Reasoning?)

Tables 1 and 2 compare AESD with the base Qwen3 models, OPSD, and OP^{2}SD across seven reasoning benchmarks. Across both model scales, AESD achieves the highest average Avg@12 accuracy and Accuracy per Token (APT), indicating that the improvement is not obtained merely by generating more tokens.

Qwen3-4B. AESD achieves an average accuracy of 53.89%, compared with 52.48% for Qwen3-4B, 52.55% for OPSD, and 52.17% for OP^{2}SD. It obtains the best or tied-best accuracy on six of

Table 2: Accuracy and token efficiency on Qwen3-8B. We report Avg@12 (%) and Accuracy per Token (APT, $\times 10^{-5}$). Higher is better; the best result on each benchmark is in **bold**.

	2025			2026				
Method	ARML	MPfG	Apex	AIME	BrUMO	CMIMC	ArXivMath	Avg.
Avg@12 (%) ↑								
Qwen	75.83	70.83	10.46	68.06	66.67	68.75	16.04	53.81
OPSD	75.83	75.00	10.28	**68.33**	69.44	**71.18**	16.67	55.25
OP^{2}SD	**77.92**	73.33	10.46	66.67	68.89	68.40	**17.50**	54.74
AESD	77.50	**77.08**	**10.82**	66.67	**72.22**	69.79	**17.50**	**55.94**
Accuracy per Token (APT, $\times 10^{-5}$) ↑								
Qwen	4.867	4.341	0.471	3.960	4.216	4.520	0.904	3.326
OPSD	5.034	4.615	0.456	4.052	4.406	**4.800**	0.925	3.470
OP^{2}SD	**5.287**	4.547	0.471	**4.102**	4.405	4.656	1.002	3.496
AESD	5.257	**4.803**	**0.491**	3.871	**4.768**	4.774	**1.014**	**3.568**

Table 3: **Does accumulated experience require a learned representation?** We compare AESD with increasingly structured targets constructed from the same experience-induced distributions.

Supervision target	Aggregation	Learned z	Avg@12 (%)	APT ($\times 10^{-5}$)
Base p_t^0	–	–	47.38	2.950
Single $q_{j,t}$	✗	✗	48.17	**3.220**
Aggregated $\bar{q}_t$	✓	✗	47.54	3.212
$(1-\alpha)p_t^0 + \alpha\bar{q}_t$	✓	✗	48.32	3.184
Proximal target	✓	✗	48.65	3.159
T_t^z (AESD)	✓	✓	**49.21**	3.121

the seven benchmarks, with particularly consistent gains across AIME, MPfG, Apex, CMIMC, and ArXivMath. AESD also achieves the highest APT on all seven benchmarks, raising the overall average from 3.323 for the base model and 3.416 for OPSD to 3.499. Thus, the accuracy improvement is accompanied by stronger output-length-normalized performance.

Qwen3-8B. The same trend holds at the larger model scale. AESD reaches 55.94% average accuracy, improving over the Qwen3-8B base model by 2.13 percentage points and over OPSD and OP^{2}SD by 0.69 and 1.20 points, respectively. The gains are not uniform across individual benchmarks: OPSD remains stronger on AIME and CMIMC, while OP^{2}SD slightly outperforms AESD on ARML. Nevertheless, AESD obtains the highest overall accuracy and the highest average APT of 3.568, with the strongest APT on MPfG, Apex, BrUMO, and ArXivMath.

Overall, these results show that AESD consistently improves the aggregate reasoning performance of both Qwen3-4B and Qwen3-8B. Notably, it does so without paired reference solutions and improves over both reference-conditioned OPSD and explicit cross-problem conditioning in OP^{2}SD, supporting accumulated experience as an effective source of self-distillation supervision.

5.3 Ablation Study (What Makes Accumulated Experience Effective?)

We analyze the key design choices behind AESD using Qwen3-4B at step 50 under the same evaluation protocol as the main experiments. The ablations follow the construction of AESD from experience-induced predictions to a learned and accumulated representation.

Experience Representation. We first ask whether historical experience should be used directly or represented through the learned target T_t^z. Starting from the single-experience distribution $q_{j,t}$, we progressively introduce experience aggregation, the proximal target, and the learned representation z. As shown in Table 3, a single historical experience already provides a useful learning signal, whereas direct aggregation does not further improve accuracy. The proximal target improves over direct aggregation, while the learned target T_t^z achieves the highest Avg@12, exceeding the arithmetic mixture and proximal target by 0.89 and 0.56 percentage points, respectively. Its APT of

Table 4: Effect of retaining the learned experience representation across training batches.

Representation z	Accumulation	Avg@12 (%)	APT ($\times 10^{-5}$)
Fixed $z = z_0$	✗	47.38	2.960
Batch-local z	✗	48.73	3.051
Batch-local z+matched extra updates	✗	48.82	3.066
Persistent z (AESD)	✓	**49.21**	**3.121**

3.121 improves over the base model but remains below the direct experience targets, indicating an accuracy gain rather than a uniform improvement on both metrics. These results suggest that AESD benefits from representing the experience-induced predictive change rather than simply averaging experience-conditioned distributions.

Cross-Batch Accumulation. Given the benefit of the learned representation, we next examine whether z should persist across training updates. We compare the persistent representation in AESD with a fixed initialization, a batch-local variant reinitialized for each batch, and a batch-local control with matched extra updates. Table 4 shows that learning a batch-local z raises Avg@12 from 47.38% to 48.73%. Matched extra updates yield 48.82% and an APT of 3.066, while persistent z reaches 49.21% and 3.121, respectively. Thus, persistence improves accuracy by 0.39 percentage points and APT by 0.055 even over the control with extra updates. The largest gain comes from learning z, with cross-batch retention providing an additional benefit beyond the tested update-budget control.

Experience Source and Structure. Finally, we examine what information in the historical experience $e_j = (x_j, \hat{y}_j)$ contributes to the signal. We remove historical experience, restrict it to the current batch, mismatch questions and responses, or disrupt response order before constructing $q_{j,t}$. Table 5 shows that intact historical experiences provide the strongest supervision on both metrics. Relative to using no experience (47.86%), current-batch experience reduces Avg@12 to 47.60%. Mismatched pairs provide a modest gain to 48.18%, whereas shuffled responses retain more of the benefit, reaching 48.43%. Intact historical pairs achieve 49.21% and the highest APT of 3.121. These results suggest that some useful signal survives disrupted pairing or response order, but preserving both gives the strongest performance among the tested variants. Taken together, these results show that historical experience is most effective when its predictive effect is represented through z and retained across training updates, rather than being used directly as an explicit supervision context.

Table 5: Effect of the source and structure of historical experience e_j.

Experience e_j	Avg@12 (%)	APT ($\times 10^{-5}$)
None	47.86	3.015
Current-batch e_j	47.60	3.005
Mismatched $(x_j, \hat{y}_{j'})$	48.18	3.034
Shuffled $(x_j, \widetilde{y}_j)$	48.43	3.072
Historical $(x_j, \hat{y}_j)$	**49.21**	**3.121**

5.4 Analysis of the Learned Experience Representation

We analyze how base regularization affects the learned target, whether the representation generalizes, and the resulting local distillation signal.

Sensitivity to Base Regularization. The learned experience target is anchored to the base prediction through the regularization term $D_{\mathrm{KL}}(T_t^z \| p_t^0)$. We vary its weight λ while keeping all other settings fixed to examine how strongly the experience-induced target should be constrained. Figure 4 shows a non-monotonic response: accuracy is 48.47% at $\lambda = 0$ (Table 12), falls at weak positive weights, and peaks at $\lambda = 0.8$. The default $\lambda = 1.0$ remains close to the optimum, indicating that AESD benefits from anchoring the experience-induced target to the base prediction without requiring a narrowly tuned value. This supports the role of base regularization in controlling the magnitude of the learned predictive shift.

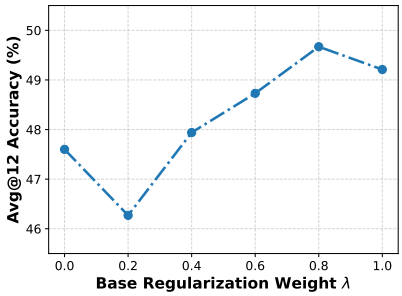

Figure 4: Sensitivity to the base-regularization weight λ.

Generalization to Unseen Problems. To test whether z merely fits the problems encountered during training, we evaluate its ability to reconstruct the aggregated experience target on held-out problems. We use 16 training problems and 32 unseen problems, while fixing the response trajectories, token positions, and sampled experiences across checkpoints. Figure 5 reports $D_{\mathrm{KL}}(\bar{q}_t \| T_t^z)$ as training progresses. The matching error decreases on both training and unseen problems. On unseen problems, the KL decreases from 0.0300 at initialization to approximately 0.0200 at step 50, a reduction of about one third, and remains below this level at subsequent checkpoints. This indicates that z captures experience-induced predictive changes that transfer beyond the problems on which it is optimized, rather than only fitting the training instances.

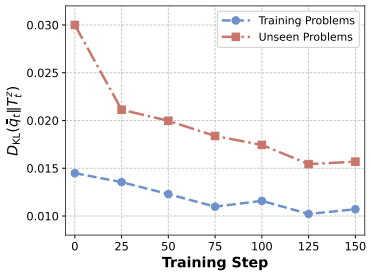

Figure 5: Generalization of the learned representation: $D_{\mathrm{KL}}(\bar{q}_t \| T_t^z)$ on training and unseen problems.

Local Distillation Signal without Pointwise Clipping. We next examine the magnitude of the supervision induced by AESD. All statistics are computed on the same held-out trajectories with the student fixed at step 50. As shown in Table 6, AESD produces a much smaller absolute target discrepancy than unclipped OPSD and OP^{2}SD. Its mean KL is 0.00137, compared with 0.15925 for OPSD and 0.05818 for OP^{2}SD. The corresponding mean logit-gradient norm is 0.00574, comparable to the magnitude obtained only after applying pointwise clipping to OPSD (0.00554) and OP^{2}SD (0.00517). Thus, AESD naturally induces a low-magnitude local distillation signal without explicit token-level clipping. This analysis concerns logit gradients on fixed trajectories rather than full parameter gradients during training.

Table 6: Local distillation signal on held-out trajectories.

Method	Mean KL	Logit-grad. norm
Without pointwise clipping		
OPSD	0.15925	0.23566
OP^{2}SD	0.05818	0.20515
AESD	**0.00137**	0.00574
After pointwise clipping		
OPSD	–	0.00554
OP^{2}SD	–	**0.00517**

5.5 Hyperparameter Sensitivity

We study the sensitivity of AESD to the number of historical experiences K and learnable soft tokens M, using $K = 2$ and $M = 8$ as the default setting. As shown in Fig. 6, performance is relatively stable across different values of M. For K, a single experience performs worse, while aggregating multiple experiences consistently improves accuracy. Overall, the default $M = 8$ and $K = 2$ provides a strong performance–cost tradeoff.

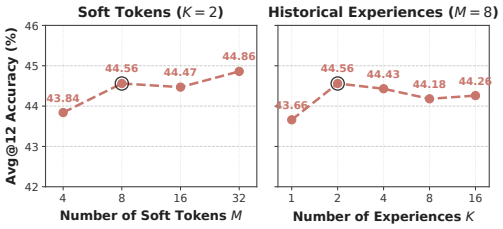

Figure 6: Sensitivity to the number of soft tokens M and historical experiences K on Qwen3-4B.

6 Conclusion

We introduced AESD, a reference-free on-policy self-distillation framework that transforms a model's accumulated historical experience into reusable training supervision. Unlike methods that rely on reference solutions, AESD focuses on how self-generated responses to other problems change the model's current predictive distribution. These changes are represented through learnable soft tokens and distilled back into the student as experience-derived targets. Experiments on Qwen3-4B and Qwen3-8B show that AESD improves overall mathematical reasoning performance. Our analysis also shows that the learned experience representation transfers to previously unseen problems, while requiring neither historical experiences nor additional soft context at inference time. AESD also suggests that a model's own past behavior can serve as a source of supervision for future learning. This provides a reference-free approach to self-improvement in settings where high-quality reference solutions cannot be continuously provided. Future work may explore how experiences can be selected, consolidated, weighted, and forgotten, enabling models to continuously learn and improve from their own interactions.

AI USE STATEMENT

We used generative AI tools solely for language polishing and assistance with writing research code. All other aspects of the research, including problem formulation, method design, theoretical analysis, experimental design and execution, and interpretation of results, were carried out by the authors. All AI-assisted text was reviewed and revised by the authors, and all AI-assisted code was inspected and tested by the authors. The authors take full responsibility for the final manuscript, implementation, and experimental results.

ETHICS STATEMENT

This work studies on-policy self-distillation for improving mathematical reasoning in large language models using existing model checkpoints, publicly available training data, and mathematical reasoning benchmarks. The study does not involve recruiting human participants or collecting new personal data. Model training and evaluation are conducted in controlled offline experiments. The research focuses on methodological development and benchmark evaluation and does not involve deployment in high-stakes settings.

REPRODUCIBILITY STATEMENT

Section 4 specifies AESD's experience aggregation, learned experience representation, and student distillation objective. Section 5.1 and Appendix A describe the models, training data, experience construction, optimization settings, benchmark subsets, and evaluation protocols. Sections 5.3–5.5 and Appendix B provide ablations and analyses of the learned representation, hyperparameter sensitivity, training cost, and transfer. AESD training alternates updates to the soft representation and the student's LoRA parameters; inference uses only the adapted student. We will publicly release the code and configuration files to facilitate reproducibility.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *ICLR*, 2024.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Lucas Caccia, Alan Ansell, Edoardo Ponti, Ivan Vulić, and Alessandro Sordoni. Training plug-n-play knowledge modules with deep context distillation. *COLM*, 2025.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. *arXiv preprint arXiv:2605.00674*, 2026.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *ICLR*, 2024.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, et al. Openthoughts: Data recipes for reasoning models. In *ICLR*, 2026.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Yuki Ichihara, Naoto Iwase, Mohammad Atif Quamar, and Junpei Komiyama. Privileged solutions or context-induced teacher behavior? dissecting on-policy self-distillation. *arXiv preprint arXiv:2608.09228*, 2026.

Dengyang Jiang, Xin Jin, Dongyang Liu, Zanyi Wang, Mingzhe Zheng, Ruoyi Du, Xiangpeng Yang, Qilong Wu, Zhen Li, Peng Gao, Harry Yang, and Steven Hoi. D-opsd: On-policy self-distillation for continuously tuning step-distilled diffusion models. *arXiv preprint arXiv:2605.05204*, 2026.

Kalle Kujanpää, Pekka Marttinen, Harri Valpola, and Alexander Ilin. Efficient knowledge injection in LLMs via self-distillation. *Transactions on Machine Learning Research*, 2025.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026a.

Yijiang Li, Bingyang Wang, Yijun Liang, Yunjie Tian, Di Fu, and Nuno Vasconcelos. On-policy self-distillation without any supervision. *arXiv preprint arXiv:2608.06296*, 2026b.

Xiaogeng Liu, Xinyan Wang, Yingzi Ma, Yechao Zhang, and Chaowei Xiao. When are teacher tokens reliable? position-weighted on-policy self-distillation for reasoning. *arXiv preprint arXiv:2605.21606*, 2026.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. https://thinkingmachines.ai/blog/on-policy-distillation.

Jingcheng Niu, Xingdi Yuan, Tong Wang, Hamidreza Saghir, and Amir H Abdi. Llama see, llama do: A mechanistic perspective on contextual entrainment and distraction in llms. In *ACL*, 2025.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Samyak Shrestha and Alexander Tessier. Rethinking privileged information in on-policy self-distillation. *arXiv preprint arXiv:2608.18271*, 2026.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Hao Wang, Guozhi Wang, Han Xiao, Yufeng Zhou, Yue Pan, Jichao Wang, Ke Xu, Yafei Wen, Xiaohu Ruan, Xiaoxin Chen, et al. Skill-sd: Skill-conditioned self-distillation for multi-turn llm agents. *arXiv preprint arXiv:2604.10674*, 2026a.

Kejing Wang, Toan Nguyen, Minh Hoang Nguyen, Simon Khan, and Flora D Salim. Road-vla: Robust online adaptation via self-distillation for vision-language-action models. *arXiv preprint arXiv:2606.25800*, 2026b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 2026a.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 2026b.

Qianhao Yuan, Jie Lou, Xing Yu, Hongyu Lin, Le Sun, Xianpei Han, and Yaojie Lu. Vision-opd: Learning to see fine details for multimodal llms via on-policy self-distillation. *arXiv preprint arXiv:2605.18740*, 2026.

Huang Yue, Wang Yanbo, Xu Zixiang, Gao Chujie, Wu Siyuan, Ye Jiayi, Chen Xiuying, Chen Pin-Yu, and Zhang Xiangliang. Breaking focus: Contextual distraction curse in large language models. *arXiv preprint arXiv:2502.01609*, 2025.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *ICML*, 2026.

ON-POLICY SELF-DISTILLATION VIA ACCUMULATED EXPERIENCE REPRESENTATION (*Supplementary Material*)

Anonymous authors
Paper under double-blind review

A ADDITIONAL EXPERIMENTAL DETAILS

A.1 TRAINING DATA AND EXPERIENCE CONSTRUCTION

We use OpenThoughts-Math-30K (Guha et al., 2026), which contains 29,434 mathematical reasoning problems. For the main training runs, we use a fixed subset of 2,400 distinct questions and access only the problem text; reference solutions and correctness labels are not used by AESD.

During training, the current student generates responses to sampled training questions with thinking disabled and a maximum generation length of 1,024 tokens. For each student trajectory, AESD samples $K = 2$ responses to different problems as cross-problem experiences. The two experiences are processed independently, and their predictive distributions are averaged rather than concatenating them into a single context. No correctness-based filtering is applied to the experience buffer.

A.2 TRAINING CONFIGURATION

Following the OPSD setting (Zhao et al., 2026), we use the same training configuration unless otherwise specified. AESD uses $M = 8$ shared learnable soft tokens. For each batch, we first update the soft representation and then recompute the distillation target to update the student's LoRA parameters.

We train all methods for 150 student updates. The student learning rate is initialized at 5×10^{-6} and linearly decayed to zero without warmup. AESD uses a fixed soft-token learning rate of 10^{-3}. All methods use LoRA with rank 64 and scaling parameter 128.

Table 7 summarizes the main training configurations.

Table 7: Training configurations used in the reported experiments.

Setting	AESD	OPSD	OP^{2}SD
Student updates	150	150	150
Effective batch size	32	32	32
Student rollouts	4,800	4,800	4,800
Rollout token limit	1,024	1,024	1,024
Student thinking	Off	Off	Off
Conditioned-branch thinking	Off	On	On
LoRA rank / scaling	64 / 128	64 / 128	64 / 128
Initial student LR	5×10^{-6}	5×10^{-6}	5×10^{-6}
LR schedule	Linear	Linear	Linear
Pointwise KL clip	None	10^{-6}	10^{-6}

The OPSD and OP^{2}SD baselines use the pointwise KL-clipping threshold of 10^{-6} used in their corresponding training configuration. The methods share the same model initialization, LoRA configuration, number of student updates, rollout-length limit, effective batch size of 32, and full-run budget of 4,800 student rollouts (150 updates at 32 rollouts each). They differ in conditioned-branch thinking and pointwise KL clipping, as shown in Table 7.

A.3 Evaluation Benchmarks

We evaluate on seven recent mathematical reasoning benchmarks from 2025–2026. Table 8 lists the subsets used in our experiments. For competitions with multiple rounds, we evaluate only the subsets specified below. The full suite contains 196 problems; the 105-problem ablation subset consists of AIME, ARML, BrUMO, and ArXivMath.

Table 8: Evaluation benchmarks and included subsets.

Benchmark	Included subset	Problems
AIME 2026	AIME I and II	30
ARML 2025	Individual and Team	20
BrUMO 2026	Division-A Individual	15
CMIMC 2026	Text-only Individual	24
Math Prize for Girls 2025	Full set	20
ArXivMath May 2026	May release	40
Apex-Shortlist 2025	Selected subset	47
Total		196

Apex-Shortlist is distinct from the smaller Apex Full benchmark. We focus on recent benchmarks to reduce reliance on long-established test sets that may be more susceptible to benchmark contamination, although recency alone does not guarantee the absence of contamination.

A.4 Evaluation Protocol and Metrics

All methods are evaluated with thinking enabled under identical decoding settings. At inference time, AESD discards both the soft representation and the experience buffer; only the adapted student model is used.

We sample 12 responses per problem with temperature 1.0 and top-p 1.0. Let $c_{ij} \in 0, 1$ indicate whether the j-th response to problem i is correct, and let ℓ_{ij} denote its number of generated tokens. For a benchmark with N problems, we report

$$\text{Avg@12} = \frac{100}{12N} \sum_{i=1}^{N} \sum_{j=1}^{12} c_{ij}, \qquad \text{APT} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{12} c_{ij}}{\sum_{i=1}^{N} \sum_{j=1}^{12} \ell_{ij}}. \tag{13}$$

Avg@12 is reported as a percentage. Accuracy per Token (APT) measures the number of correct predictions per generated token and is reported in units of 10^{-5}. All generated tokens are included in the denominator. The Avg. columns in Tables 1 and 2 are unweighted means of the seven benchmark-level values. The checkpoint curves in Fig. 7 instead show question-weighted Avg@12 and pooled APT over the full 196-problem suite.

A.5 Evaluation Schedule

We evaluate every 25 updates, from step 25 through step 150. Table 9 records the checkpoints underlying the main results for each method and model size. Each reported checkpoint is evaluated on all seven benchmarks, and both metrics are computed from its generated responses. The ablations in Sec. 5.3 are evaluated at step 50 on Qwen3-4B for all variants.

Table 9: Reported training steps.

Model	OPSD	OP^{2}SD	AESD
Qwen3-4B	25	125	100
Qwen3-8B	25	125	100

B Additional Experiments

B.1 Performance across Training Checkpoints

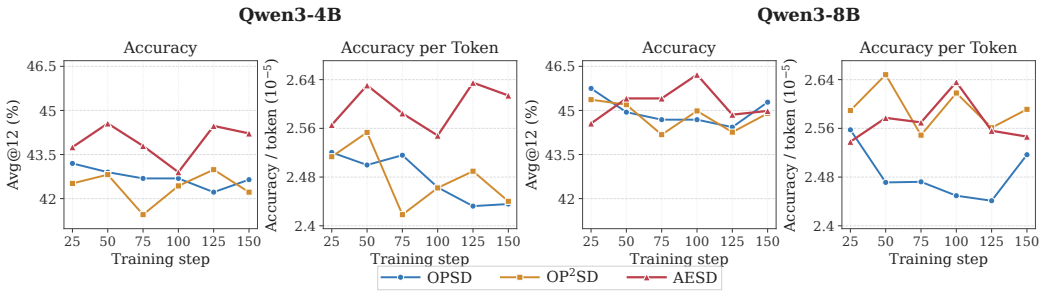

Figure 7: Accuracy and token efficiency across training checkpoints for Qwen3-4B and Qwen3-8B. The panels show question-weighted Avg@12 and pooled APT over the evaluation suite; these differ from the unweighted benchmark averages in Tables 1 and 2.

Figure 7 shows test performance at the regularly evaluated training steps. For Qwen3-4B, AESD has the highest plotted accuracy and APT at every evaluated step, although accuracy declines between steps 50 and 100 before recovering. For Qwen3-8B, AESD reaches its highest accuracy at step 100; the relative ordering of the methods varies at other steps. Accuracy and APT need not peak at the same checkpoint. The curves describe the evolution of both metrics over training under the evaluation schedule in Appendix A.

B.2 Representation Persistence and Optimizer State

Table 10: **Separating representation persistence from optimizer-state persistence.**

Variant	Persistent z	Persistent opt. state	Avg@12 (%)	APT ($\times 10^{-5}$)
Batch-local	✗	✗	48.73	3.051
Persistent z, reset optimizer	✓	✗	49.10	3.110
Persistent z + optimizer	✓	✓	**49.21**	**3.121**

Table 10 separates the effect of retaining the soft representation from that of retaining its optimizer state. Keeping z while resetting the optimizer improves Avg@12 from 48.73% to 49.10% and APT from 3.051 to 3.110 relative to the batch-local variant. Retaining the optimizer state as well gives a smaller additional gain, reaching 49.21% and 3.121. Thus, most of the improvement in this comparison remains when optimizer state is reset, supporting persistence of the representation itself as the main contributor. This complements the matched-extra-update control in Table 4.

B.3 Content of Historical Experience

Table 11: **What information in historical experience matters?** Non-solution controls are matched in effective context length.

Experience context	Avg@12 (%)	APT ($\times 10^{-5}$)
None	47.86	3.015
Current-batch experience	47.60	3.005
Mismatched $(x_j, \hat{y}_{j'})$	48.18	3.034
Shuffled $(x_j, \widetilde{y}_j)$	48.43	3.072
Generic explanatory context	47.74	3.012
Mathematical non-solution context	48.05	3.026
Historical $(x_j, \hat{y}_j)$	**49.21**	**3.121**

Table 11 compares historical experience with non-solution contexts matched in effective length. Generic explanatory context reaches 47.74% Avg@12 and 3.012 APT, slightly below the no-experience control (47.86% and 3.015). Mathematical non-solution context reaches 48.05% and 3.026, but still falls short of intact historical question–response pairs (49.21% and 3.121). Historical experience therefore exceeds the mathematical non-solution control by 1.16 percentage points. Together with the disrupted-pair and shuffled-order controls, these results suggest that the observed benefit cannot be attributed solely to adding a similarly sized explanatory or mathematical context.

B.4 Initialization Calibration and Base Regularization

Table 12: **Disentangling initialization calibration and base regularization.**

Target	Init. calibration	Base reg.	Avg@12 (%)	APT ($\times 10^{-5}$)
q_t^z	✗	✗	46.92	2.925
q_t^z	✗	✓	48.12	3.035
T_t^z	✓	✗	48.47	3.081
T_t^z (AESD)	✓	✓	**49.21**	**3.121**

Table 12 examines the two components used to construct and constrain the learned target. Using the uncalibrated q_t^z without base regularization yields 46.92% Avg@12 and 2.925 APT. Adding regularization increases these values to 48.12% and 3.035; applying initialization calibration alone yields 48.47% and 3.081. Combining both components gives the best results, 49.21% and 3.121, exceeding calibration alone by 0.74 percentage points and regularization alone by 1.09 points. Both components therefore contribute in this comparison: calibration removes the initial soft-token shift, while base regularization constrains the subsequently learned target.

B.5 Evolving Experience and Representation Accumulation

Table 13: **Interaction between evolving experience and representation accumulation.**

Experience	Evolves	Persistent z	Avg@12 (%)	APT ($\times 10^{-5}$)
Static	✗	✗	47.72	2.998
Static	✗	✓	48.35	3.052
Evolving	✓	✗	48.71	3.084
Evolving	✓	✓	**49.21**	**3.121**

Table 13 compares static and evolving experience sources with and without a persistent representation. For static experience, retaining z raises Avg@12 from 47.72% to 48.35% and APT from 2.998 to 3.052. With evolving experience, persistence raises accuracy from 48.71% to 49.21% and APT from 3.084 to 3.121. Evolving experience also improves both metrics at either persistence setting. The combination achieves the best observed result, indicating that updating the experience source and retaining the learned representation each contribute under these controls; accumulation is useful even when the experience source is static.

B.6 Matched Teacher-Side Configurations

Table 14: **Controlled comparison under matched teacher-side configurations.**

Method	Cond. thinking	KL clipping	Avg@12 (%)	APT ($\times 10^{-5}$)
OPSD (reported config.)	On	10^{-6}	47.62	3.054
OPSD (matched)	Off	None	48.05	3.096
OP^{2}SD (reported config.)	On	10^{-6}	47.54	3.072
OP^{2}SD (matched)	Off	None	48.18	3.105
AESD	Off	None	**49.21**	**3.121**

Table 14 controls for two teacher-side configuration differences: conditioned-branch thinking and pointwise KL clipping. Disabling both raises OPSD accuracy from 47.62% to 48.05% and OP^{2}SD accuracy from 47.54% to 48.18%, while also improving their APT. AESD still achieves higher accuracy by 1.16 and 1.03 percentage points, respectively, and the highest APT of 3.121, compared with 3.096 and 3.105. Thus, AESD's advantage remains under the tested matched settings. Because thinking and clipping are changed together, this comparison does not isolate their individual effects.

B.7 Practical Training Cost

Table 15 reports measured wall-clock training cost on the same four-H200 node after ten warm-up steps. AESD takes 26.00 seconds per step, compared with 23.80 for OPSD and 18.44 for OP^{2}SD. These measurements correspond to approximately 9.2% and 41.0% higher time per step, respectively, or 8.67 minutes for 20 AESD updates. The precision settings differ: AESD uses FP32, whereas both baselines use BF16. These results therefore describe the practical cost of the reported implementations, rather than isolating algorithmic overhead at matched precision. This training cost is distinct from the inference-time APT metric; AESD removes the experience buffer and soft tokens at inference time.

Table 15: Measured training cost on Qwen3-4B.

Method	Precision	20 Steps	Time / Step
OPSD	BF16	7.93 min	23.80 s
OP^{2}SD	BF16	**6.15 min**	**18.44 s**
AESD	FP32	8.67 min	26.00 s

B.8 Transfer to a New Student

Table 16: **Transfer of the learned experience representation to a new student.**

Supervision on $\mathcal{D}_B$	Representation	Avg@12 (%)	APT ($\times 10^{-5}$)
Base target p_t^0	None	47.38	2.950
Transferred $T_t^{z_A}$	Frozen from $\mathcal{D}_A$	48.44	3.048
In-domain $T_t^{z_B}$	Learned on $\mathcal{D}_B$	**49.05**	**3.102**

Table 16 evaluates whether a representation learned on $\mathcal{D}_A$ can provide supervision to a new student trained on $\mathcal{D}_B$. Using the frozen transferred representation z_A improves Avg@12 from 47.38% to 48.44% and APT from 2.950 to 3.048 relative to the base target. Learning z_B on $\mathcal{D}_B$ performs best, reaching 49.05% and 3.102. Transfer therefore retains a useful part of the improvement, while leaving a 0.61-percentage-point accuracy gap to the in-domain representation. This supports reuse of the learned predictive effect across the reported training settings, while showing the additional benefit of adapting it to the destination data.

B.9 Uncertainty of Performance Differences

Table 17 reports differences in Avg@12 relative to AESD, with 95% confidence intervals. Differences and intervals are expressed in percentage points; a negative difference indicates lower accuracy than AESD.

Table 17: **Uncertainty of performance differences.** Negative values indicate lower accuracy than AESD.

Comparison	Δ Avg@12	95% CI
OPSD − AESD	-1.59	$[-3.05, -0.14]$
OP^{2}SD − AESD	-1.67	$[-3.14, -0.22]$
U-OPSD − AESD	-1.10	$[-2.20, -0.06]$
Proximal − AESD	-0.56	$[-1.32, 0.18]$
Batch-local − AESD	-0.48	$[-1.18, 0.16]$

The intervals for OPSD, OP^{2}SD, and U-OPSD lie below zero. The intervals for the proximal target and batch-local representation include zero.

B.10 Evaluation Prompt

The following prompt is used for mathematical reasoning evaluation, with the question text inserted at `{current problem}`.

Evaluation prompt

```
Problem: {current problem}

Please reason step by step, and put your final answer
within \boxed{}.
```

C Limitations

Our experiments focus on mathematical reasoning with Qwen3-4B and Qwen3-8B, and it remains unclear how well the observed behavior extends to other model families, scales, and task domains. AESD also uses a fixed soft-token budget and a simple historical experience buffer throughout training. More adaptive mechanisms for controlling representation capacity, selecting useful experiences, and discarding stale or redundant ones remain to be explored. Finally, our current analysis primarily studies predictive effects at the token-distribution level; a more detailed understanding of what semantic or reasoning information is retained in the learned experience representation is left for future work.