BEYOND SOLVABILITY: TASK LEARNABILITY AS A STATIC PRIOR FOR LLM RL POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) has become a central post-training paradigm for eliciting reasoning capabilities in large language models, yet uniform task sampling allocates compute without regard to differences in how tasks respond to optimization. Existing task-valuation methods mostly rely on snapshot-based signals such as current pass rate or reward, which estimate how solvable a task is under the current policy. However, tasks with similar current solvability can still differ substantially in how positively they respond to further training. We study this residual axis as *task learnability*: a regime-conditional measure of expected positive response to continued training under a fixed RL post-training regime. By analyzing per-task reward trajectories, we find that learnability is reproducible across independently sampled training contexts and predictive of downstream utility. To make this signal practical before training begins, we propose TRAJVAL, a probe-based estimator that approximates per-task learnability from a short probe run and two endpoint evaluations. TRAJVAL can be used either as a standalone static prior for task sampling or as a multiplicative prior for existing online schedulers. Experiments on mathematical and logical reasoning benchmarks across multiple model scales show that TRAJVAL improves data efficiency over uniform sampling and provides complementary gains when combined with online scheduling methods.

1 INTRODUCTION

Reinforcement learning (RL) has become a central post-training paradigm for eliciting reasoning capabilities in large language models (Guo et al., 2025; Jaech et al., 2024; Yang et al., 2025). A recurring concern in this setting is data efficiency: training pools contain large numbers of tasks whose contributions to learning differ substantially, yet uniform sampling allocates compute without regard to this heterogeneity. A growing body of work on *task-value estimation* addresses this by weighting or scheduling tasks during training to accelerate convergence (Yu et al., 2025; Gao et al., 2025; Shen et al., 2025; Zheng et al., 2025; Wu et al., 2026). These methods predominantly rely on the model's instantaneous performance as the primary indicator of task value. By monitoring metrics such as pass rates or rewards at the current training state, they seek to prioritize tasks based on their immediate difficulty or "solvability".

However, current solvability provides only a partial view of task value. Two tasks with similarly low pass rates at the current policy state can still differ sharply in how they respond to continued optimization. In this work, we study this residual axis as *learnability*: a task's expected positive response to continued training under a fixed post-training regime. Importantly, we do not treat learnability as an immutable property of a task in isolation. Rather, it is a *regime-conditional* quantity, defined with respect to a given model family, initialization, optimization algorithm, reward function, data source, and training budget.

This notion is complementary to *solvability*. Solvability asks how well the current policy already performs on a task; learnability asks whether that task is likely to improve productively under additional training. The distinction matters operationally: among tasks with similar current solvability, some continue to yield positive reward response under training, while others remain flat, saturated, or even regress. Existing scheduling methods mainly target the former quantity, which can leave useful information about future training response untapped.

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.

In this work, we study **learnability** as a complementary dimension of task value. By analyzing training traces, we observe that the policy exhibits highly heterogeneous learning responses to different tasks: while performance on some tasks reliably improves as training progresses, performance on others stagnates or degrades. Empirical analysis demonstrates that this divergence is both context-robust and predictive of RL post-training gains: each task's learnability reproduces across independent training runs sampled from the same data distribution, and yields measurable improvements when leveraged to select or weight tasks.

Directly measuring this signal is expensive: it requires tracking each task's performance trajectory over a full post-training run that sweeps the entire dataset for multiple epochs. To turn learnability into a prior usable before RL post-training begins, we apply two simplifications: summarizing the trajectory by a few performance values along it, and replacing the full post-training run with a short proxy run used to obtain those values. Building on these two reductions, we propose **TRAJVAL**, a preprocessing-stage estimator that yields a continuous sampling prior. It can serve as a standalone method for task weighting and composes multiplicatively with existing online task-scheduling methods.

We evaluate TRAJVAL on mathematical and logical reasoning benchmarks across multiple model scales. Empirical results show that it consistently outperforms uniform sampling and provides stable improvements over competitive task-value baselines when used as a plug-in prior. Our code is available at an anonymous repository: https://anonymous.4open.science/r/Task-Learnability-EB5E.

Our contributions are threefold:

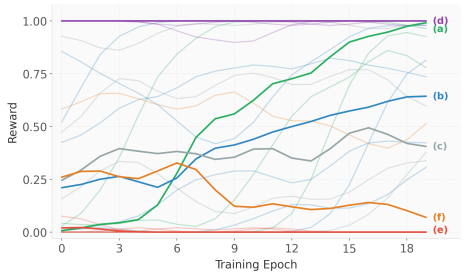
- We distinguish *learnability* from snapshot-based *solvability* in RL post-training, and operationalize learnability as a task's regime-conditional positive response to continued training.
- Empirical evidence shows that *learnability* is reproducible across independently sampled contexts and predictive of downstream utility, even among tasks with matched initial solvability.
- We propose TRAJVAL, a lightweight probe-based estimator of learnability that can be used either as a standalone static prior or as a multiplicative prior for existing online task schedulers.

2 RELATED WORK

Static Task-Value Curation. Offline task-value curation scores the data pool before training. In supervised fine-tuning, methods such as LIMA, LIMO, and s1 select high-value subsets using base-model pass rates or external model assessments (Zhou et al., 2023; Ye et al., 2025; Muennighoff et al., 2025). Similar ideas have been explored in RL using signals such as base-model uncertainty (Zhao et al., 2025), complexity (Chen et al., 2025b), and difficulty tiers (Parashar et al., 2025). These approaches rely on a fixed snapshot of the initial model and therefore provide limited information about how a task will respond to continued optimization. LIMR (Li et al., 2025a) moves toward trajectory-aware valuation, but it requires a full training pass over the pool.

Dynamic Task-Value Scheduling. Dynamic schedulers instead estimate task value online as training progresses. Representative approaches use rollout-based pass-rate signals or model-based approximations such as value models and Bayesian posteriors (Yu et al., 2025; Yue et al., 2025; Gao et al., 2025; Shen et al., 2025; Qu et al., 2026), while others filter prompts using reward consistency or single-step policy improvement signals (Zheng et al., 2025; Wu et al., 2026; Gu et al., 2026; Chen et al., 2025a). These methods provide adaptive feedback from the current policy state, but mainly emphasize instantaneous task solvability.

Learning Dynamics and Our Positioning. The use of training dynamics to characterize samples has a long history in supervised learning. Dataset Cartography (Swayamdipta et al., 2020) and EL2N (Paul et al., 2021), for example, show that the evolution of loss or confidence under training reveals information about sample difficulty, learnability, and generalization impact that is not captured by single-state evaluation. Our work brings this trajectory-based perspective to RL post-training by focusing on a lightweight estimate of task learnability that complements existing state-based scheduling signals.



Profile	Criteria	Behavior	Learn.
(a) STABLE	$\Delta{>}0.4, R^2{>}0.4$ or $\sigma{\leq}0.35$	Steady rise	✓
(b) SLUGGISH	$\Delta{\in}(0.15, 0.4]$ or $\sigma{>}0.35$	Gradual rise	✓
(c) INEFFECTIVE	$\Delta{\in}(-0.05, 0.15]$, $\sigma{\leq}0.35$	Flat / noisy	✓
(d) MASTERED	$\bar{r}_{\text{early}}{>}0.9$, $\Delta{<}0.1$	Near-ceiling	✗
(e) UNLEARNED	$\bar{r}_{\text{late}}{<}0.1$, $\Delta{<}0.1$	Near-floor	✗
(f) FORGETTING	$\Delta{<}-0.05$	Net decline	✗

Figure 1: Representative per-task reward trajectories from the diagnostic pool $\mathcal{D}$ over 20 training epochs. Bold lines highlight six coarse trajectory patterns used for diagnosis. The right panel summarizes the corresponding profiles. The profile distribution over $\mathcal{D}$ is shown in Figure 6(a).

3 A LEARNABILITY-BASED VIEW OF TASK VALUE IN RL POST-TRAINING

In this section, we develop a learnability-based view of task value in two stages. We first examine how each task's reward evolves over the course of RL training, and show that these per-task trajectories motivate *learnability* as a task-level signal distinct from snapshot-based solvability (§3.1); we then show that this signal is both **reproducible across independent training contexts drawn from the same data distribution** and **predictive of downstream utility** (§3.2).

3.1 HETEROGENEOUS PER-TASK LEARNABILITY UNDER TRAINING

To examine per-task reward trajectories, we uniformly sample a diagnostic pool $\mathcal{D}$ of 2,048 mathematical reasoning tasks and train Qwen3-1.7B with GRPO for 20 epochs under a fixed training recipe (prompt batch size 64, 16 rollouts per prompt; full details in Appendix A). As Figure 1 shows, tasks with similar initial pass rates can follow sharply different reward trajectories under continued training. This heterogeneity is temporal: it is not visible from a single snapshot of current pass rate alone.

This observation motivates our central distinction. A task's *solvability* describes the current policy's performance on that task, whereas its *learnability* describes how positively the task responds to continued optimization under a fixed regime. Intuitively, tasks that exhibit a steady, sustained upward trend with substantial reward growth exhibit high learnability, as they provide consistent and informative optimization signals. Conversely, tasks whose trajectories are highly volatile, stagnant, or declining exhibit low learnability, suggesting that the current optimization regime either fails to extract constructive signals from these tasks or struggles to consolidate the resulting updates.

To visualize this heterogeneity, we summarize each full reward trajectory using three simple descriptors—reward gain Δ, standard deviation σ, and linear-fit R^2—and use them to define six coarse trajectory profiles (Figure 1). These profiles are intended purely as a **diagnostic lens**: they help summarize representative trajectory shapes, but they are *not* the formal definition of learnability, nor are they meant to form an exhaustive taxonomy of all possible training dynamics.

For descriptive analysis, we group profiles with non-negative training response (e.g., STABLE, SLUGGISH, and INEFFECTIVE) separately from saturated, low-response, or regressive profiles (e.g., MASTERED, UNLEARNED, and FORGETTING). This binary grouping is used only for diagnosis in this section. In Section 4, we will replace these coarse categories with a continuous score used for actual data allocation.

3.2 LEARNABILITY IS CONTEXT-ROBUST AND PREDICTIVE OF TRAINING VALUE

We next validate whether the learnability signal suggested by these trajectories satisfies two properties required for practical use: (*i*) whether it is reproducible across independently sampled training contexts under a fixed regime, and (*ii*) whether it predicts downstream utility for data allocation. We provide empirical evidence for both using Qwen3-1.7B with GRPO.

Cross-context reproducibility. We construct two probe sets $\mathcal{P}_1$ and $\mathcal{P}_2$ (each of size 512, uniformly sampled from $\mathcal{D}$, with 256 shared tasks) and independently train on $\mathcal{D}$, $\mathcal{P}_1$, and $\mathcal{P}_2$ for 20 epochs

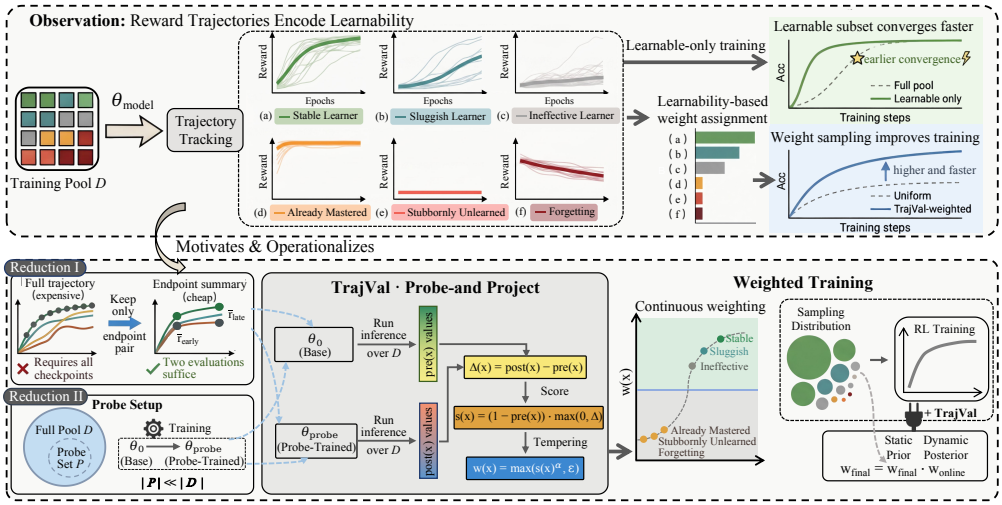


Figure 2: Overview of TRAJVAL. Top: per-task reward trajectories reveal heterogeneous training response beyond initial solvability. Bottom: TRAJVAL approximates this response using two reductions: from dense trajectories to an endpoint-based summary, and from full-pool training to a lightweight probe run. The resulting score is used as a static prior for prior-weighted sampling in RL training, either alone or in combination with an online scheduler.

under a *fixed training recipe* (same model, algorithm, and hyperparameters). Despite differences in mixture and the inherent stochasticity of RL, the macroscopic profile distributions remain nearly identical across the three contexts (Figure 6(a)–(c) in Appendix B.2). At the individual level, the 256 shared tasks receive consistent profile assignments: Cohen's κ reaches **0.776** across the six profiles and **0.879** under the binary learnable grouping (Figure 6(d)–(e)). The same pattern also appears under Llama-3.2-3B (Appendix B.3), supporting the view that learnability is a **context-robust task-level signal** rather than an artifact of any particular mixture.

Predictive value for downstream utility. We next ask whether the learnability grouping translates into downstream utility under two complementary operationalizations, measured by average accuracy across six mathematical reasoning benchmarks (details in Appendix A.2). *As a hard cut*, restricting training to tasks marked *learnable* ($\approx 41\%$ of $\mathcal{D}$) matches the full pool's converged accuracy (≈ 0.41) in $3.6\times$ **fewer steps** (Figure 3, left). *As a soft prior over the full pool*, an oracle-based scheme that assigns higher sampling probability to learnable profiles—without discarding any task—outperforms random sampling at matched training budget after both have converged (0.410 vs 0.387, Figure 3, right). The signal therefore yields measurable gains under both pruning-style and weighting-style uses, suggesting that its utility is not specific to either form of data allocation. In Section 5.3, we further isolate learnability from initial solvability through a controlled analysis.

Together, these results suggest that learnability behaves as a reproducible and practically useful dimension of task value under a fixed training regime. The remaining challenge is computational: obtaining this signal at oracle quality requires a full T-epoch training pass over $\mathcal{D}$, which largely offsets the efficiency gains it can provide. Section 4 therefore asks whether a much cheaper probe procedure can preserve the task ordering most relevant to learnability-aware data allocation.

4 TRAJVAL: A PROBE-BASED ESTIMATOR OF LEARNABILITY

Section 3 used full reward trajectories to characterize an oracle notion of learnability. TRAJVAL does not redefine that notion; rather, it provides a much cheaper **estimator** of it. The key question is therefore not whether a small number of evaluations can reconstruct every detail of the full trajectory, but whether they preserve the task ordering most relevant to learnability-aware data allocation.

To make oracle learnability practical before the main RL run begins, TRAJVAL applies two reductions. First, it replaces dense trajectory tracking with an endpoint-based summary of training response.

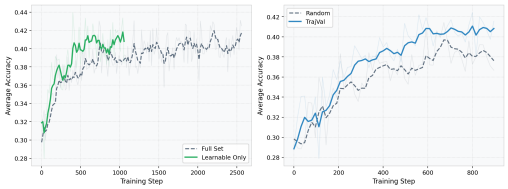


Figure 3: Effect of learnability-based data allocation on Qwen3-1.7B with GRPO, averaged over six mathematical reasoning benchmarks. **Left:** Training on the learnable subset versus the full pool over 20 epochs. **Right:** An oracle-based soft prior over the full pool versus random sampling at matched training budget.



Figure 4: All tasks in $\mathcal{D}$ projected onto the slope–R^2 space of their training trajectories. Colors indicate profile labels derived from the endpoint pair $(\bar{r}_{\text{early}}, \bar{r}_{\text{late}})$.

Second, it replaces full-pool training with a much cheaper probe training run on a small uniformly sampled subset. Together, these reductions yield a static learnability prior that can be computed offline and used either on its own or in combination with existing online schedulers.

4.1 Lightweight Estimation

Reduction I: From Trajectories to Endpoint-Based Summaries. Continuous checkpoint evaluation is expensive. We therefore ask whether the task ordering relevant to learnability can be captured by a much smaller summary: the mean rewards over an early phase and a late phase, denoted $\bar{r}_{\text{early}}$ and $\bar{r}_{\text{late}}$, together with their difference $\Delta = \bar{r}_{\text{late}} - \bar{r}_{\text{early}}$. This summary no longer preserves the full trajectory shape, but it still retains the minimal temporal contrast needed to distinguish a task's current solvability from how it responds over training. The goal is thus not to reconstruct the entire reward curve, but to preserve the distinctions most relevant to learnability-aware allocation.

Using this endpoint pair, we derive a simplified endpoint-only approximation to the diagnostic profiles in Figure 1, bypassing statistics such as σ and R^2 that require intermediate checkpoints. To test whether this summary retains the global structure of trajectory behavior, we project all tasks in $\mathcal{D}$ onto the slope–R^2 space computed from their full reward sequences, which is independent of the endpoint approximation. As Figure 4 shows, labels derived solely from the endpoint pair remain well separated in this endpoint-independent space, indicating that the summary preserves the distinctions most relevant to learnability estimation.

Reduction II: From Full-Pool Training to Probe Training. Even with the endpoint summary, estimating $\bar{r}_{\text{late}}$ from full-pool training would still require a complete T-epoch pass over $\mathcal{D}$. We replace this with a much cheaper proxy. Let $\mathcal{P} \subset \mathcal{D}$ be a small uniformly sampled probe set with $|\mathcal{P}| \ll |\mathcal{D}|$. Because $\mathcal{P}$is drawn from the same training distribution, optimizing the base model θ_0 on $\mathcal{P}$ exposes the policy to a representative subset of the optimization signals present in $\mathcal{D}$. The resulting probe-trained model θ_{probe} can therefore serve as a practical proxy for a later training state.

This substitution is empirical rather than axiomatic, so we verify it directly. For each task $x \in \mathcal{D}$, we compare probe-derived endpoint estimates against oracle values obtained from the full 2,048-task profiling run. The Spearman correlations are $\rho = 0.940$ for $\bar{r}_{\text{early}}$ and $\rho = 0.876$ for $\bar{r}_{\text{late}}$, indicating that the relative ordering needed for learnability-aware allocation largely survives this substitution.

4.2 Implementation: From Probe Estimators to Sampling Weights

Given probe-based estimates of the early and late rewards for every task $x \in \mathcal{D}$, we now convert them into a sampling distribution. Concretely, we replace the coarse profile-based diagnostic view from Section 3.1 with a continuous learnability score, as illustrated in Figure 2 and Algorithm 1.

Endpoint Estimation. For each task $x \in \mathcal{D}$, we summarize the probe trajectory by averaging per-task accuracies over K checkpoints from an early window and a late window:

$$\bar{r}_{\phi}(x) \;=\; \frac{1}{K}\sum_{t \in \mathcal{T}_{\phi}} \text{acc}(x, \theta_t), \quad \phi \in \{\text{early}, \text{late}\} \tag{1}$$

The net improvement is then captured by their difference, $\Delta(x) = \bar{r}_{\text{late}}(x) - \bar{r}_{\text{early}}(x)$. The window size K is a preprocessing-budget hyperparameter that trades additional inference passes for reduced rollout variance. In our main experiments, we evaluate the most economical setting $K = 1$; any gain observed there carries over to higher-K settings with lower estimator noise.

Continuous Learnability Scoring. To convert the probe statistics into per-task sampling weights, we define a simple *learnability-aware task-value proxy* that combines two desiderata from §3.1: tasks should still have room to improve, and they should exhibit a positive response under training:

$$s(x) \;=\; \underbrace{\big(1 - \bar{r}_{\text{early}}(x)\big)}_{\text{learning headroom}} \cdot \underbrace{\max\big(0,\ \Delta(x)\big)}_{\text{directional alignment}}. \tag{2}$$

The directional term suppresses tasks whose reward remains flat or declines under probe training, while the headroom term down-weights tasks that are already near saturation. Their product therefore favors tasks that are both still improvable and already showing evidence of constructive response.

The role of the headroom term is not to replace $\Delta(x)$ as the indicator of improvement, but to encode an allocation preference: among tasks with comparable positive gain, higher priority is assigned to those with greater remaining room for improvement under the main training budget. Accordingly, the score places a modest penalty on near-mastered tasks whose short-term gains may be positive but whose residual training value is limited. Additional intuition for this score design is provided in Appendix B.1.

Sampling Integration and Diversity. Scores $s(x)$ can span a wide range, and direct proportional sampling risks over-concentrating on a narrow high-score subset. To retain learnability-aware priority while preserving distributional coverage, we apply power compression and a probability floor:

$$w(x) \;=\; \max\big(s(x)^{\alpha},\ \epsilon\big), \quad p(x) \;=\; \frac{w(x)}{\sum_{x' \in \mathcal{D}} w(x')}. \tag{3}$$

The exponent $\alpha < 1$ compresses the score range, limiting the weight ratio between high- and low-scoring tasks. The floor ϵ ensures that every task retains a minimal sampling probability, including those with near-zero learnability scores—a deliberate choice that maintains exposure to the full data distribution and guards against degenerate concentration.

Integration as a Multiplicative Prior. TRAJVAL provides a static learnability prior that composes with existing online schedulers. For a method with dynamic sampling weight $w_{\text{online}}(x, t)$, we use

$$w_{\text{final}}(x, t) = w_{\text{TRAJVAL}}(x) \cdot w_{\text{online}}(x, t). \tag{4}$$

This combination is complementary in spirit: TRAJVAL supplies a stable offline prior from initialization, while the online scheduler adapts to the evolving policy state during training. TRAJVAL therefore targets a failure mode that online methods do not directly address, namely poor allocation before sufficient online evidence has accumulated.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

We validate TRAJVAL across two domains, three model variants spanning different scales and families, and multiple evaluation benchmarks. Full implementation details are provided in Appendix A.

Tasks and Datasets. We evaluate across two distinct domains: **mathematical reasoning**, which requires rigorous multi-step numerical deduction, and **logical reasoning**, which demands structured, abstract inference over diverse puzzle formats. We train on **DAPO-Math** (Yu et al., 2025) (17,398 tasks) for math, and the **Logic subset** (4,998 tasks) of the GURU (Cheng et al., 2025) dataset for logic, which spans ordering puzzles, graph puzzles, ARC-AGI (Chollet et al., 2024; 2025), and BARC (Li et al., 2024) tasks, with exact-match binary reward.

Evaluation and Metrics. For **math**, we assess on six benchmarks: MATH500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), AIME24, AIME25, OlympiadBench (He et al., 2024), and

	Math						Logic					
	Qwen3-1.7B			Qwen3-4B			Qwen3-1.7B			Qwen3-4B		
Method	S2B ↓	Best Acc ↑	AUC ↑	S2B ↓	Best Acc ↑	AUC ↑	S2B ↓	Best Acc ↑	AUC ↑	S2B ↓	Best Acc ↑	AUC ↑
Random (BASE-GRPO)	–	0.3967	0.3620	–	0.5060	0.4829	–	0.1826	0.1544	–	0.2052	0.1932
Random+TRAJVAL	65.66	**0.4105**	**0.3730**	60.00	**0.5190**	**0.4931**	73.91	**0.1904**	**0.1663**	40.00	**0.2260**	**0.2101**
BOTS Shen et al. (2025)	–	0.4042	0.3648	–	0.5076	0.4767	–	0.1862	0.1556	–	0.2216	0.2076
BOTS+TRAJVAL	95.00	**0.4081**	**0.3703**	75.00	**0.5178**	**0.4863**	95.65	**0.1866**	**0.1645**	60.87	**0.2232**	**0.2112**
GRESO Zheng et al. (2025)	–	0.4028	0.3621	–	0.5062	0.4695	–	0.1878	0.1568	–	0.2244	0.2082
GRESO+TRAJVAL	90.00	**0.4083**	**0.3681**	75.00	**0.5170**	**0.4794**	86.96	**0.1914**	**0.1648**	78.26	**0.2252**	**0.2103**

Table 1: Main results across Math and Logic domains at two model scales. TRAJVAL is used either as a standalone static prior over uniform sampling or as a multiplicative prior for online schedulers.

AMC23. For **logic**, we evaluate on a held-out set from Ordering Puzzles and ARC-AGI (Chollet et al., 2024). We report three complementary metrics per benchmark, as well as their macro-average (**Avg**): (1) **Best Acc**, the peak sliding-window mean accuracy; (2) **AUC**, the time-averaged accuracy across all training steps; and (3) **Steps-to-Baseline (S2B)**, the fraction of steps required for a TRAJVAL-augmented method to first match the Best Acc of its corresponding baseline (lower is better). All results are means over 2 seeds.

Implementation. All experiments are built on the Trinity-RFT (Pan et al., 2025) framework with **GRPO** (Shao et al., 2024). Our main results use **Qwen3-1.7B** and **Qwen3-4B** (Yang et al., 2025), while **Llama-3.2-3B** (Grattafiori et al., 2024) is included as an additional cross-family validation in Appendix B.3. In both domains, RL is applied with $n = 16$ rollouts per prompt. For TRAJVAL, the learnability prior is derived from two inference passes over the base model θ_0 and a lightweight probe-trained model θ_{probe}; the sampling weight is parameterized by a power-compression exponent $\alpha = 0.3$ and a smoothing floor $\epsilon = 0.05$.

Baselines. We compare against three representative task-selection strategies: (1) **BASE-GRPO** (Shao et al., 2024), vanilla GRPO with uniform task sampling; (2) **BOTS** (Shen et al., 2025), a Bayesian online scheduler that maintains posterior estimates of per-task difficulty; and (3) **GRESO** (Zheng et al., 2025), a pre-rollout filtering method that skips temporally uninformative prompts. Baseline-specific settings follow the original works and are detailed in Appendix A.5.

5.2 MAIN RESULTS

We evaluate TRAJVAL across two reasoning domains and three base training strategies. Table 1 summarizes the main results, and Figure 12 shows the corresponding evaluation curves. The overall pattern supports the central claim of the paper: task allocation for RL post-training benefits from modeling not only how solvable a task currently is, but also how positively it is likely to respond to continued optimization. Per-benchmark results are provided in Appendix C.3, and Appendix C.1 gives rollout-level views of training dynamics.

TrajVal as a standalone method. As a direct replacement for uniform sampling, TRAJVAL consistently improves both convergence speed and peak performance over BASE-GRPO across all four domain–scale configurations. The gains are visible in both Math and Logic, with especially large acceleration on Logic at 4B, where the method reaches the baseline peak in only 40% of the steps.

TrajVal as a plug-in prior. TRAJVAL also yields consistent gains when composed with existing online schedulers. Augmenting BOTS and GRESO consistently reduces S2B and improves AUC across configurations, with Best Acc gains in most settings. This pattern supports the view that learnability provides a stable prior that complements online solvability estimates, particularly when those estimates are still noisy early in training.

Consistency across domains and scales. The improvements of TRAJVAL persist across both reasoning domains and model scales. We further verify the same qualitative trend on an additional model family, Llama-3.2-3B, with supporting cross-family evidence in Appendix B.3. We next provide a controlled analysis to directly test the central claim that this gain cannot be reduced to current solvability alone.

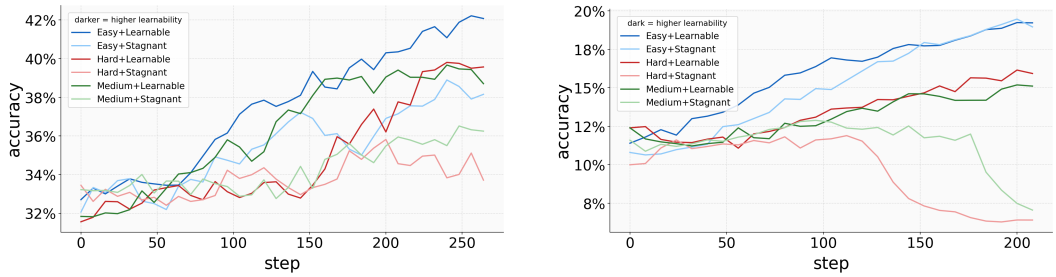


Figure 5: Within-pre-bin evaluation accuracy curves on Qwen3-1.7B with GRPO for high-Δ and low-Δ sub-pools: Math (left) and Logic (right).

		Math				Logic			
Tercile	**Sub-pool**	Avg pre	Avg Δ	Best Acc ↑	AUC ↑	Avg pre	Avg Δ	Best Acc ↑	AUC ↑
Low	low-Δ	0.0015	0.0140	0.338	0.3379	0.0000	0.0026	0.124	0.0946
	high-Δ	0.0016	0.1080	**0.434**	**0.3545**	0.0000	0.0818	**0.152**	**0.1367**
Mid	low-Δ	0.1660	0.0570	0.361	0.3449	0.0065	0.0401	0.134	0.1114
	high-Δ	0.1660	0.2860	**0.391**	**0.3675**	0.0065	0.1384	**0.150**	**0.1325**
High	low-Δ	0.6810	0.0480	0.387	0.3586	0.4695	0.2994	0.194	0.1550
	high-Δ	0.6820	0.2000	**0.398**	**0.3813**	0.4693	0.4578	**0.198**	**0.1635**

Table 2: Within-pre-bin analysis on Qwen3-1.7B across Math and Logic. Matched Avg pre within each tercile isolates the effect of Δ on downstream performance.

5.3 Within-Pre-Bin Analysis: Learnability Beyond Initial Solvability

A central claim of this work is that learnability is not reducible to current solvability. We test this with a controlled within-bin analysis that isolates the effect of $\Delta(x)$ after stratifying by $\bar{r}_{\text{early}}(x)$. If high-$\Delta$ tasks still yield better downstream outcomes than low-Δ tasks within the same pre-bin, then the predictive signal cannot be explained by initial solvability alone.

Setup. Within each domain, we partition the training pool into three pre-bins of equal size by ranking tasks according to their pre values and splitting them into the low, mid, and high pre groups, corresponding to the bottom, middle, and top thirds of the ranked list, respectively. Within each bin, tasks are split by $\Delta(x)$ into two equal-size sub-pools—**high-Δ** (top 50%) and **low-Δ** (bottom 50%)—with average pre matched between the two, so that the only systematic difference is in Δ. This controlled setup is intended to isolate the signal quality of $\Delta(x)$, rather than to replicate TrajVal's weighted sampling scheme.

Results. Table 2 and Figure 5 show that across all six bin–domain combinations, **high-Δ sub-pools uniformly outperform their low-Δ counterparts** in both Best Acc and AUC, despite matched average pre within each bin. The effect is strongest in the Low-pre Math bin, where high-Δ tasks reach 0.434 Best Acc versus 0.338 for low-Δ tasks. On Math, meaningful gaps persist in the Mid and High bins; on Logic, the same monotonic ordering holds across all terciles. This shows that tasks with the same current solvability can still differ substantially in training value, and that this residual difference is captured by learnability.

5.4 Ablation and Practical Analysis

We further examine TrajVal along six complementary dimensions:

Score composition. All three single-dimension variants—**pre-only**, **post-only**, and **Δ-only**—underperform full TrajVal, especially in convergence speed on Math. The results in Table 3 indicate that headroom and directional alignment are complementary: pre captures remaining room for improvement, while Δ captures responsiveness to optimization. Only their combination prioritizes tasks that are both improvable and productively learnable under the current training regime.

Method	Math (Avg)			Logic		
	S2B ↓	Best Acc ↑	AUC ↑	S2B ↓	Best Acc ↑	AUC ↑
Random	–	0.3967	0.3620	–	0.1826	0.1544
Score composition						
pre-only	85.86	0.4084	0.3675	79.08	0.1904	0.1625
post-only	85.86	0.4067	0.3662	82.61	0.1897	0.1622
Δ-only	85.86	0.4086	0.3660	78.26	0.1864	0.1641
Probe set size						
$\lvert\mathcal{P}\rvert = 256$	100.00	0.3968	0.3623	**72.73**	**0.1916**	0.1639
$\lvert\mathcal{P}\rvert = 1024$	90.91	0.4021	0.3670	73.91	0.1892	0.1638
TrajVal ($\lvert\mathcal{P}\rvert = 512$)	**65.66**	**0.4105**	**0.3730**	73.91	0.1904	**0.1663**

Table 3: Ablation results on Qwen3-1.7B over score composition and probe set size.

Diagnostic analyses. Appendix B.4 provides two additional findings. First, $s(x)$ concentrates sampling weight on learnable tasks (67% of weight for 41% of the pool), and the score distribution remains nearly unchanged across independent probe sets. Second, the score ranking stabilizes early: the Spearman correlation with the final ranking reaches $\rho \geq 0.85$ by probe epoch $k{=}14$ out of $T{=}20$ (Figure 9), suggesting the probe horizon can be shortened without changing score quality.

Hyperparameter sensitivity. We examine two key hyperparameters. A moderate probe set size ($|\mathcal{P}|{=}512$, roughly 3% of the pool) in Table 3 provides the best overall trade-off. Across $\alpha \in \{0.1, 0.3, 0.5\}$, all settings improve over RANDOM (Appendix B.7), indicating that TRAJVAL is not highly sensitive to the exact choice of α.

Computational overhead. TRAJVAL adds a one-time probe run on $\approx 3\%$ of the pool and two inference passes for endpoint estimation, while the per-step sampling overhead is negligible ($\approx 0.01\%$ of total pipeline time; Appendix B.5).

Cross-regime reuse. We further examine whether a TRAJVAL prior estimated under one regime can be reused across model scales or families, rather than re-estimated for each target model (Appendix B.6). Transfer retains useful signal, particularly within the same family, while in-regime estimation achieves the strongest overall performance.

Temporal validity of the static prior. The fixed prior remains informative as the policy evolves (Appendix B.8). Although overall learnability declines as headroom is consumed, groups ranked by initial prior score stay separated throughout training: at training end, the highest-scoring group retains roughly twice the learnability of the zero-score group. High-score groups also decline faster under TRAJVAL than under BASE-GRPO, consistent with prioritized sampling consuming their headroom sooner. The static prior thus keeps identifying relatively learnable tasks throughout training.

Taken together, these results show that TRAJVAL is well-behaved in practice: its gains rely on the joint use of headroom and directional response, its ranking remains stable under lightweight probing, its performance is robust to hyperparameter variation, its static prior remains effective over a long training horizon, and its one-time preprocessing cost is amortized through cross-model prior reuse.

6 CONCLUSION

We introduced learnability as a regime-conditional dimension of task value in RL that complements current solvability, and showed that tasks with similar initial pass rates can differ markedly in their response to continued optimization. To make this signal practical, we proposed TRAJVAL, a probe-based estimator that converts early/late endpoint statistics from a short probe run into sampling weights for main RL training. Across different tasks, models, and utilized settings, TRAJVAL improves data efficiency over uniform sampling and adds gains when combined with online schedulers. More broadly, our results suggest that task valuation for RL post-training should consider not only how solvable a task is now, but also how productively it is likely to improve under further training, a perspective that may also inform future adaptive priors.

AI USE STATEMENT

In this work, generative AI tools were used only as supplementary aids for language polishing, grammar correction, and improving the clarity of exposition in text that was first drafted by the authors, as well as for routine, non-novel implementation tasks (e.g., boilerplate code, plotting scripts, and refactoring). We have not used generative AI tools to generate research ideas, formulate the core hypotheses, design the method, produce or analyze experimental results, or draw scientific conclusions; these were carried out entirely by the authors.

ETHICS STATEMENT

All authors have read and adhered to the ICLR 2027 Code of Ethics. This work introduces TRAJVAL, a method for improving the efficiency of reinforcement fine-tuning of Large Language Models, and does not involve human subjects, animal experiments, or the processing of personally identifiable information. All datasets used in our experiments are publicly available and well-established benchmarks within the research community, and all software, datasets, and frameworks used are released under the permissive Apache-2.0 open-source license. Our goal is to make AI research more sustainable and accessible, and we do not foresee any direct negative societal impacts or ethical concerns arising from TRAJVAL. The authors declare no conflict of interest.

REPRODUCIBILITY STATEMENT

We are committed to ensuring the full reproducibility of our research. To this end, we will release our source code, which includes the implementation of TRAJVAL and the scripts required to reproduce all experiments reported in this paper. The appendix provides a comprehensive description of our experimental setup, including all model configurations, dataset processing steps, and hyperparameter settings.

REFERENCES

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970*, 2025a.

Zhuoyue Chen, Jihai Zhang, Ben Liu, Fangquan Lin, and Wotao Yin. Scale down to speed up: Dynamic data selection for reinforcement learning. *Training*, 2500:3000, 2025b.

Zhoujun Cheng, Shibo Hao, Tianyang Liu, Fan Zhou, Yutao Xie, Feng Yao, Yuexin Bian, Yonghao Zhuang, Nilabjo Dey, Yuheng Zha, et al. Revisiting reinforcement learning for llm reasoning from a cross-domain perspective. *arXiv preprint arXiv:2506.14965*, 2025.

Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. Arc prize 2024: Technical report. *arXiv preprint arXiv:2412.04604*, 2024.

Francois Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. Arc-agi-2: A new challenge for frontier ai reasoning systems. *arXiv preprint arXiv:2505.11831*, 2025.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Zhaolin Gao, Joongwon Kim, Wen Sun, Thorsten Joachims, Sid Wang, Richard Yuanzhe Pang, and Liang Tan. Prompt curriculum learning for efficient llm post-training. *arXiv preprint arXiv:2510.01135*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Zhengyao Gu, Jonathan Light, Raul Astudillo, Ziyu Ye, Langzhou He, Henry Peng Zou, Wei Cheng, Santiago Paternain, Philip S Yu, and Yisong Yue. Actor-curator: Co-adaptive curriculum learning via policy-improvement bandits for rl post-training. *arXiv preprint arXiv:2602.20532*, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Wen-Ding Li, Keya Hu, Carter Larsen, Yuqing Wu, Simon Alford, Caleb Woo, Spencer M Dunn, Hao Tang, Michelangelo Naim, Dat Nguyen, et al. Combining induction and transduction for abstract reasoning. *arXiv preprint arXiv:2411.02272*, 2024.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. Limr: Less is more for rl scaling. *arXiv preprint arXiv:2502.11886*, 2025a.

Yu Li, Zhuoshi Pan, Honglin Lin, Mengyuan Sun, Conghui He, and Lijun Wu. Can one domain help others? a data-centric study on multi-domain reasoning via reinforcement learning. *arXiv preprint arXiv:2507.17512*, 2025b.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Xuchen Pan, Yanxi Chen, Yushuo Chen, Yuchang Sun, Daoyuan Chen, Wenhao Zhang, Yuexiang Xie, Yilun Huang, Yilei Zhang, Dawei Gao, et al. Trinity-rft: A general-purpose and unified framework for reinforcement fine-tuning of large language models. *arXiv preprint arXiv:2505.17826*, 2025.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, et al. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *arXiv preprint arXiv:2506.06632*, 2025.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34: 20596–20607, 2021.

Yun Qu, Qi Wang, Yixiu Mao, Vincent Tao Hu, Björn Ommer, and Xiangyang Ji. Can prompt difficulty be online predicted for accelerating rl finetuning of reasoning models? In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 1240–1250, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024.

Qianli Shen, Daoyuan Chen, Yilun Huang, Zhenqing Ling, Yaliang Li, Bolin Ding, and Jingren Zhou. Bots: A unified framework for bayesian online task selection in llm reinforcement finetuning. *arXiv preprint arXiv:2510.26374*, 2025.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9275–9293, 2020.

Jiahao Wu, Ning Lu, Shengcai Liu, Kun Wang, Yanting Yang, Li Qing, and Ke Tang. Train at moving edge: Online-verified prompt selection for efficient rl training of large reasoning model. *arXiv preprint arXiv:2603.25184*, 2026.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. In *Second Conference on Language Modeling*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Murong Yue, Wenlin Yao, Haitao Mi, Dian Yu, Ziyu Yao, and Dong Yu. Dots: Learning to reason dynamically in llms via optimal reasoning trajectories search. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yang Zhao, Kai Xiong, Xiao Ding, Li Du, Zhouhao Sun, Jiannan Guan, Wenbin Zhang, Bin Liu, Dong Hu, Bing Qin, et al. Ufo-rl: Uncertainty-focused optimization for efficient reinforcement learning data selection. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Haizhong Zheng, Yang Zhou, Brian R Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021, 2023.

Appendix

A Details of Experimental Setup

A.1 Experimental Tasks and Datasets

We validate our proposed method across two distinct domains to demonstrate its versatility: mathematical reasoning and logical reasoning. The former assesses the model's capacity for rigorous, multi-step numerical computation, while the latter evaluates its ability to perform structured, abstract reasoning across diverse puzzle types.

Mathematics For mathematical reasoning, the objective is to enhance the model's problem-solving capabilities. We use the **DAPO-Math-17k** dataset (Yu et al., 2025) for training. This dataset consists of 17,398 problems, primarily from competitive mathematics, with integer-based answers. The verified and binary reward structure (1.0 for a correct answer, 0.0 otherwise) provides a clear, objective signal for improving logical and computational accuracy.

Logic For logical and abstract reasoning, the objective is to improve the model's capacity for structured inference and pattern recognition across diverse reasoning formats. Training data is sourced from the **GURU** (Cheng et al., 2025) benchmark collection, a well-curated, cross-domain RL dataset. Specifically, we use the **Logic subset** (approximately 5.0k problems), which spans several complementary reasoning tasks: **Ordering Puzzles** (relational ordering under constraints), **Graph Puzzles** (implicit graph traversal and path reasoning), the public training splits of **ARC-AGI** (Chollet et al., 2024) and **ARC-AGI-2** (Chollet et al., 2025) (abstract grid transformation tasks), and a 3.4k-sample subset from **BARC** (Li et al., 2024) (synthetic ARC-style tasks for data augmentation). For all logic tasks, predictions are extracted from `<answer>` tags and evaluated via exact match, yielding a binary reward of 1.0 for a correct prediction and 0.0 otherwise.

A.2 Evaluation and Metrics

A.2.1 Evaluation Datasets.

Mathematics. We evaluate mathematical reasoning on a suite of benchmarks commonly used for assessing Qwen-Math series (Yang et al., 2024) models. This selection covers a wide spectrum of difficulty, from challenging high-school problems to graduate-level and Olympiad-level questions:

- **AIME24 / AIME25**: The American Invitational Mathematics Examination (AIME) features highly challenging, competition-level problems for high-school students.
- **AMC23**: A benchmark consisting of 40 problems from the American Mathematics Competitions (AMC). This widespread contest series for high-school students serves as a key test of foundational mathematical reasoning.
- **MATH500** (Hendrycks et al., 2021): A classic and challenging benchmark comprising 500 problems from high-school math competitions, designed to test multi-step reasoning across algebra, combinatorics, geometry, number theory.
- **Minerva** (Lewkowycz et al., 2022): A benchmark featuring 272 advanced problems from university and graduate-level STEM courses, covering subjects like differential equations and special relativity.
- **OlympiadBench** (He et al., 2024): An Olympiad-level benchmark containing 675 exceptionally difficult problems in mathematics and science that demand profound and complex reasoning abilities.

Logic We evaluate logical and abstract reasoning on a held-out evaluation set of 500 problems, drawn from two complementary benchmarks that probe distinct reasoning faculties:

- **Ordering Puzzles**: A set of relational reasoning problems that require the model to infer and satisfy ordering constraints among entities, serving as a probe for general structured logical inference.

- **ARC-AGI** (Chollet et al., 2024): A benchmark of grid-based abstract reasoning tasks requiring the model to identify latent transformation rules from input–output examples and apply them to novel grids.

A.2.2 EVALUATION METRICS.

All reported scores are averaged over 2 independent random seeds. A sampling temperature of $T = 1.0$ is used throughout for generation (Li et al., 2025b; Cheng et al., 2025). To ensure robust estimates on stochastic benchmarks, we generate multiple solutions per problem for AIME 24/25 (8 generations) and AMC23 (4 generations). Online evaluation is performed every 12 steps for Mathematics and every 8 steps for Logic. Training curves are visualized with an Exponential Moving Average (EMA, factor 0.6) applied solely for display purposes; all quantitative metrics are computed on raw trajectories.

We report three complementary metrics for each benchmark as well as their macro-average across all benchmarks (**Avg**):

- **Best Acc**: The peak accuracy achieved over the full training run, computed as the maximum value of a sliding-window mean with a window of 5 consecutive evaluation steps applied to the raw accuracy curve. This windowed estimator reduces sensitivity to stochastic fluctuations at individual checkpoints. For the **Avg** column in Mathematics, Best Acc is the peak of the macro-averaged accuracy curve, computed by first averaging the seed-averaged per-benchmark trajectories across all benchmarks at each step and then applying the same sliding-window procedure.
- **AUC**: The time-averaged accuracy across the entire training run, computed as the arithmetic mean of all evaluation-step accuracies on the seed-averaged trajectory. This metric captures the overall training efficiency of a method, rewarding approaches that not only converge to high accuracy but do so consistently throughout training, rather than merely at the final checkpoint.
- **Steps-to-Baseline (S2B)**: A data-efficiency metric defined pairwise for each TRAJVAL-augmented method relative to its corresponding baseline. Let T denote the Best Acc of the baseline and s_{base} the training step at which the baseline's sliding-window mean first attains T. S2B is then the first step s_{comp} at which the augmented method's sliding-window mean reaches T, expressed as a percentage of s_{base}:

$$\text{S2B} = \frac{s_{\text{comp}}}{s_{\text{base}}} \times 100\%.$$

  Lower values indicate that the augmented method matches the baseline's best performance with fewer training steps. For the **Avg** column in Mathematics, S2B is computed on the macro-averaged curve across all benchmarks.

A.3 TRAINING DETAILS

Framework and Infrastructure. All experiments are conducted on the **Trinity-RFT** (Pan et al., 2025) framework, utilizing nodes equipped with 8 NVIDIA A100 (80GB) GPUs. We employ PyTorch's Fully Sharded Data Parallel (FSDP) (Zhao et al., 2023) for efficient distributed training. During the exploration phase, response generation is accelerated using vLLM (Kwon et al., 2023) via flash-attention (Dao et al., 2022). We utilize GRPO (Shao et al., 2024) as the RL algorithm. We set *enable_thinking* as *False* for Qwen3 series to avoid the long thinking paradigm.

Domain-Specific Training Processes. Our experimental design validates our method across two distinct domains—Mathematics and Logic—spanning three model variants to assess generality across both model scale and model family: **Qwen3-1.7B**, **Qwen3-4B** (Yang et al., 2025), and **Llama-3.2-3B** (Grattafiori et al., 2024).

• For **Mathematics**, we perform RL directly on the **instruction-tuned** variants of all three base models. The reward mechanism is binary: a response is awarded a reward of 1.0 if the answer extracted from the `\boxed{}` environment matches the ground truth, and 0.0 otherwise. This verification is performed using the `math-verify` library. We employ a relatively large number of rollouts per prompt ($n = 16$) to provide stable variance estimates for policy gradient updates.

• For **Logic**, we likewise perform RL directly on the **instruction-tuned** variants of all three models, without any preceding supervised fine-tuning stage. The reward signal is binary and based on exact

match: a response receives a reward of 1.0 if the prediction extracted from the <answer> tags exactly matches the ground-truth output, and 0.0 otherwise. We adopt the same number of rollouts per prompt ($n = 16$) as in the Mathematics setting, maintaining consistency across domains.

Prompt. We employ domain-specific prompts tailored to the structure of each task type. In Mathematics, a consistent prompt format is applied to both training and evaluation. For Logic tasks, distinct prompt templates are used for each sub-task category (ARC-style grid tasks, ordering puzzles, and graph puzzles) to match the corresponding input format, with the same templates applied during both training and evaluation. The exact templates are shown below.

Mathematics Training & Evaluation Prompt

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

Logic-ARC Training & Evaluation Prompt

```
System_Prompt: You are a world-class puzzle solver with exceptional pattern
recognition skills. Your task is to analyze puzzles, spot patterns, and provide
direct solutions. Given input-output grid pairs as reference examples, carefully
observe the patterns to predict the output grid for new test input. Each pair
follows the same transformation rule. Grids are 2D arrays. Here are the input and
output grids for the reference examples: {User_Prompt}. What is the output grid?
Please put your answer within <answer> and </answer> tags, your final answer should
be only the output grid (2d array).
User_Prompt: {task_text}
```

Logic-OrderingPuzzle Training & Evaluation Prompt

```
System_Prompt: Solve the following puzzle to determine the order of the {User_Prompt}
from left to right. {User_Prompt}. Please put your answer within <answer> and
</answer> tags, for example <answer> ['pigeon', 'sparrow', 'quail'] </answer>.
User_Prompt: {category} and {task_text}
```

Logic-GraphPuzzle Training & Evaluation Prompt

```
System_Prompt: Given the following list of predicates: {User_Prompt}. Respond
with only the trait of the next step. Please put your answer within <answer> and
</answer> tags, for example <answer> fdebme </answer>.
User_Prompt: {task_text}
```

Hyperparameters. All three model variants share identical training hyperparameters within each domain; differences arise only between domains due to their distinct input length distributions and dataset sizes. Specifically, Logic tasks require substantially longer prompt contexts owing to the structured, multi-example format of ARC-style inputs, whereas Mathematics tasks demand longer response budgets to accommodate extended chain-of-thought reasoning. A comprehensive summary is provided in Table 4.

A.4 Policy Optimization Algorithm: GRPO

We adopt **Group Relative Policy Optimization (GRPO)** (Shao et al., 2024) as our policy gradient algorithm. A defining characteristic of GRPO is its elimination of a learned value function: rather than relying on a separate critic network to estimate baselines, it constructs the advantage signal directly from intra-group reward comparisons. Concretely, the policy parameters θ are optimized via

	Mathematics (RL)	Logic (RL)
Base Model	Qwen3-{1.7B, 4B}. / Llama-3.2-3B.	Qwen3-{1.7B, 4B}. / Llama-3.2-3B.
Common Training Parameters		
Optimizer	AdamW	AdamW
Learning Rate	1×10^{-6}	1×10^{-6}
Weight Decay	0.1	0.1
Gradient Clipping	1.0	1.0
Batch Size (Prompts)	64	64
Total Epochs	1	3
RL-Specific Parameters		
Rollouts per Prompt (n)	16	16
GRPO Clip Ratio (ϵ)	0.2	0.2
Rollout Temperature	1.0	1.0
System and Memory Parameters		
Max Prompt Length	4,096	28,672
Max Response Length	8,192	4,096
Rollout Engine Number	4	4
Evaluation Steps Interval	12	8

Table 4: Key hyperparameters for RL training across domains. All three model variants (Qwen3-1.7B, Qwen3-4B, Llama-3.2-3B) share the same configuration within each domain.

a clipped surrogate objective of the form:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{y \sim \pi_{\theta_{\text{old}}}} \Big[\min\big(r_\theta(y) A(y), \\ \text{clip}(r_\theta(y), 1-\epsilon, 1+\epsilon)\, A(y)\big) \Big]. \tag{5}$$

where $r_\theta(y) = \pi_\theta(y|x)/\pi_{\theta_{\text{old}}}(y|x)$ is the probability ratio measuring how the likelihood of response y shifts between the updated and reference policies, and the clipping operator constrains this ratio to the interval $[1-\epsilon,\ 1+\epsilon]$, bounding each gradient step within a trusted region to prevent destabilizing policy updates.

The advantage $A(y_k)$ in Equation 5 is derived through a group normalization procedure. For each prompt x, a group of G responses $Y = \{y_1, \ldots, y_G\}$ is sampled from the old policy $\pi_{\theta_{\text{old}}}$, yielding a corresponding set of rewards $\{R(y_k)\}_{k=1}^{G}$. The advantage of each response is then defined as its reward standardized by the group statistics:

$$A(y_k) = \frac{R(y_k) - \mu_Y}{\sigma_Y}, \tag{6}$$

where μ_Y and σ_Y denote the mean and standard deviation of rewards within the group, respectively. This normalization ensures that the advantage signal reflects a response's relative quality among its peers, providing a low-variance baseline without requiring any additional learned components.

A.5 BASELINES

To evaluate TRAJVAL, we select three representative online task-valuation methods as baselines. For each, we report results both with and without TRAJVAL applied as a sampling-weight multiplier. All baselines share the same base training configuration described in Appendix A.3; only the task-selection logic differs. Baseline-specific hyperparameters are set according to the recommendations of the respective original works.

- **BASE-GRPO** (Shao et al., 2024): The vanilla GRPO algorithm with uniform random task sampling, serving as the primary reference point. Tasks are drawn **randomly** from the training pool without any curriculum or difficulty-aware selection. This baseline establishes the performance floor that all task-selection methods, including TRAJVAL, are expected to improve upon.
- **BOTS** (Shen et al., 2025): **B**ayesian **O**nline **T**ask **S**election is a unified framework that recasts dynamic task selection as a Bayesian inference problem over evolving model capabilities. In

our experiments, BOTS is initialized with pre-computed pass rates from two reference models (`Qwen2.5-7B` and `Qwen3-32B`) as the initial difficulty features. We follow the recommended configuration from the original work, with the key task-selector hyperparameters set as: group size $m = 16$, implicit evidence weight $\lambda = 0.1$, regularization coefficient $\rho = 0.1$, target difficulty $p^* = 0.5$, and Thompson sampling temperature $\tau = 0.5$.

- **GRESO** (Zheng et al., 2025): **G**RPO with **E**fficient **S**elective **R**ollout is an online pre-rollout filtering algorithm motivated by the empirical observation of strong temporal consistency in prompt informativeness: prompts that yield near-zero advantage in one training epoch are likely to remain uninformative in subsequent epochs. We adopt the recommended configuration from the original paper, with the key task-selector hyperparameters set as: easy-skipping probability $p_{\text{easy}} = 0.5$, hard-skipping probability $p_{\text{hard}} = 0.5$, zero-variance target $\sigma^2_{\text{zero}} = 0.25$, probability update step $\Delta p = 0.01$, and probability bounds $p_{\min} = 0.05$, $p_{\max} = 0.95$.

B Extended Experiments and Analyses

B.1 Design Intuition of Learnability Score

Equation 2 is intended as a low-cost operationalization of the two empirical dimensions introduced in Section 3.1: *capacity to improve* and *constructive directional response*. The factor $1 - \bar{r}_{\text{early}}(x)$ down-weights tasks that are already near saturation, while $\max(0, \Delta(x))$ suppresses tasks whose reward trajectories are flat or regressive under optimization.

The multiplicative form reflects a conservative conjunctive design: a task should receive a high score only when it satisfies both conditions simultaneously. If either dimension is weak—for example, a task is already nearly mastered, or its trajectory shows little positive response—the overall score is reduced accordingly. This behavior matches the qualitative motivation in Figure 1, where tasks with meaningful training value tend to exhibit both remaining headroom and a non-negative optimization trend.

We do not claim this product form is optimal. Rather, it provides a simple and stable proxy for prioritizing tasks that are both unsaturated and empirically responsive to training, while avoiding additional estimation complexity. The ablations in Section 5.4 further support this choice by showing that combining the two factors is more effective than using either one alone.

B.2 Stability of Learnability Profiles

This subsection provides the full visualization for the cross-context stability study summarized in Section 3.2. We train Qwen3-1.7B for 20 epochs on the full pool $\mathcal{D}$ and on two independently sampled subsets $\mathcal{P}_1$ and $\mathcal{P}_2$, each of size 512, with 256 shared tasks. For each training context, we assign every task a trajectory profile using the criteria in Figure 1.

Figure 6(a)–(c) examines stability at the pool level. Despite the substantial difference in set size and the independent sampling of $\mathcal{P}_1$ and $\mathcal{P}_2$, the profile proportions remain highly similar across all three contexts. In particular, the aggregate mass of learnable profiles and non-learnable profiles changes only slightly, and the relative ordering of the major categories is preserved. This supports the view that learnability composition is largely a property of the underlying data source, rather than a fragile consequence of a specific training mixture.

Figure 6(d)–(e) examines stability at the level of individual tasks. Among the 256 shared tasks, most assignments lie on or near the diagonal of the confusion matrices. Agreement is already strong for the full six-class taxonomy, with most disagreement occurring between neighboring learnable categories such as Stable, Sluggish, and Ineffective. After collapsing profiles into the binary learnable/non-learnable partition, the consistency becomes even stronger, yielding $\kappa = 0.879$. This pattern is expected: fine-grained labels are more sensitive to small variations in trajectory shape, whereas the coarser partition captures the directional distinction that matters most for data allocation.

Taken together, the figure supports two practical conclusions used by TrajVal. First, a uniformly sampled subset can preserve the broad learnability composition of the full pool. Second, per-task learnability is sufficiently stable across contexts to serve as a reusable signal for probe-based estimation.

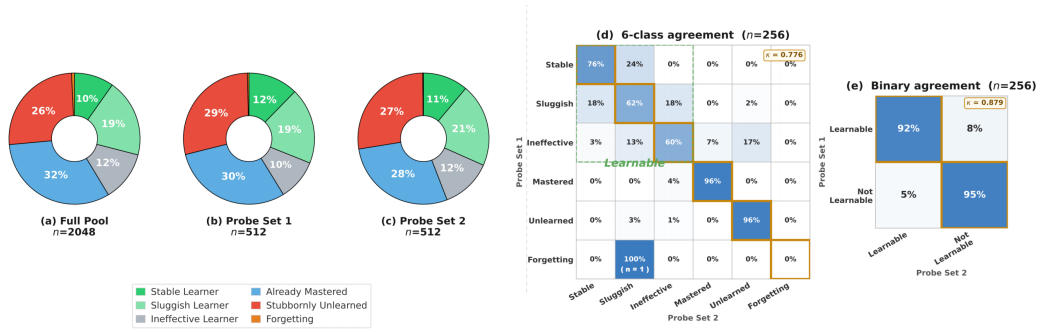


Figure 6: Qwen3-1.7B: Stability of learnability profiles across independently sampled training contexts. **(a)–(c):** The macroscopic distribution of profiles remains nearly identical across $\mathcal{D}$, $\mathcal{P}_1$, and $\mathcal{P}_2$, confirming that uniform subsampling preserves the learnability composition of the full pool. **(d)–(e):** Confusion matrices for the 256 shared tasks, evaluated under $\mathcal{P}_1$ versus $\mathcal{P}_2$. Strong per-task agreement is observed at both the fine-grained six-profile level ($\kappa = 0.776$) and the binary learnable/non-learnable level ($\kappa = 0.879$).

B.3 Learnability Stability Across Model Families on Llama

To assess whether the observations in Sections 3–5.2 extend beyond the Qwen family, we repeat the key analyses on Llama-3.2-3B. The goal of this subsection is not to introduce a new setting, but to verify that the main empirical patterns remain visible under a different base model family.

Cross-context stability. We first examine whether learnability remains stable across training contexts, as in the Qwen-based study of Section 3.2. Figure 7 shows that the macroscopic profile distribution remains similar across independently sampled training contexts, and that the profile assignment of shared tasks is stable at both the six-profile level and the binary learnable/non-learnable level. The resulting agreement scores, $\kappa = 0.762$ for the six-way taxonomy and $\kappa = 0.872$ for the binary grouping, are close to those observed in the main text. This suggests that learnability remains a measurable and reasonably stable task property under a different model family.

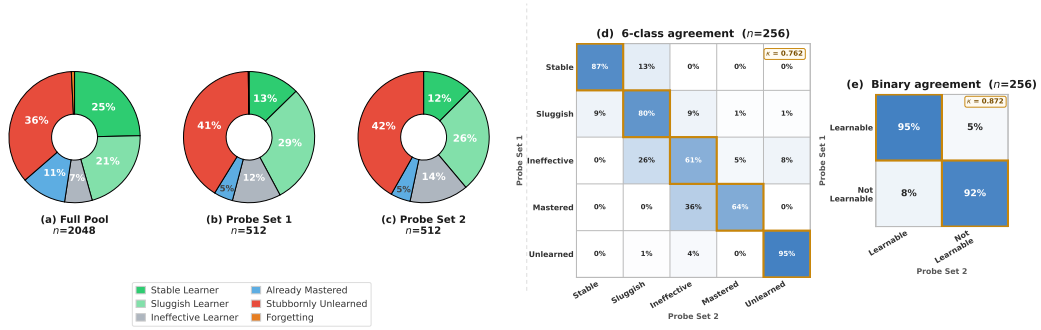


Figure 7: Llama-3.2-3B: Stability of learnability profiles across independently sampled training contexts. The configuration is the same as in Figure 6.

Learnability structure. We next revisit whether the structural organization of learnability observed in Section 3 also appears under Llama-3.2-3B. Figure 8 reproduces the slope–R^2 projection on the full 2,048-task pool, with profile labels derived from the endpoint pair $(\bar{r}_{\text{early}}, \bar{r}_{\text{late}})$. The same broad geometric structure is preserved: positively trending profiles occupy the high-slope region, while non-learnable profiles remain concentrated near low or negative slopes. This indicates that the endpoint summary continues to recover the global organization of trajectory behavior under Llama-3.2-3B, consistent with the reduction in Section 4.

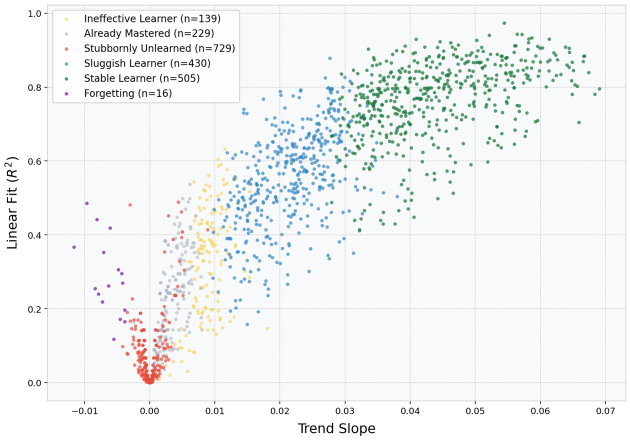


Figure 8: Llama-3.2-3B: All tasks in $\mathcal{D}$ projected onto the slope–R^2 space of their reward trajectories, colored by profile labels derived from the endpoint pair $(\bar{r}_{\text{early}}, \bar{r}_{\text{late}})$. The same broad structural separation observed in Figure 4 is preserved, indicating that endpoint-derived labels continue to recover the global organization of trajectory behavior under a different model family.

End-to-end transfer. Finally, we ask whether the practical benefits of the learnability prior also transfer to downstream RL training. Table 5 reports the end-to-end results on mathematical and logical reasoning. The overall pattern is directionally consistent with the main text: TRAJVAL usually improves either convergence speed, final accuracy, or AUC when paired with Random or BOTS, and remains particularly helpful on Logic. At the same time, the gains are smaller and less uniform than those observed for Qwen, with one Math configuration showing a regression when combined with GRESO. We therefore view the Llama results as supportive rather than definitive: they suggest that the learnability prior transfers across model families, while also indicating that its interaction with specific schedulers can remain model-dependent.

	Math (Avg)			Logic		
Method	S2B (%) ↓	Best Acc ↑	AUC ↑	S2B (%) ↓	Best Acc ↑	AUC ↑
Random	–	0.1724	0.1582	–	0.1204	0.1031
Random+TrajVal	**94.12**	**0.1800**	**0.1637**	100.00	**0.1234**	**0.1059**
BOTS	–	0.1752	0.1585	–	0.1332	0.1055
BOTS+TrajVal	**100.00**	**0.1760**	**0.1595**	**96.15**	**0.1340**	**0.1094**
GRESO	–	0.1765	**0.1646**	–	0.1252	0.1013
GRESO+TrajVal	**90.00**	**0.1791**	0.1634	**65.00**	**0.1288**	**0.1102**

Table 5: End-to-end results on mathematical and logical reasoning with Llama-3.2-3B. S2B (%) denotes the fraction of baseline training steps required to match the baseline's peak performance, where lower is better. Best Acc is the peak evaluation accuracy over the full training run, and AUC is the area under the evaluation curve.

B.4 DIAGNOSTIC ANALYSIS OF TRAJVAL SCORES

Score distribution across learnability profiles. To verify that the continuous score $s(x)$ assigns weight in a manner consistent with the qualitative learnability taxonomy introduced in Section 3, we stratify $\mathcal{D}$ by trajectory profile and report per-profile score statistics under two independently sampled probe sets $\mathcal{P}_1$ and $\mathcal{P}_2$ in Table 6. Three patterns are evident. First, the learnable profiles collectively receive 67% of total sampling weight despite constituting only 41% of the pool, representing a $1.6\times$ redirection of sampling budget toward tasks with positive learnability. Second, score statistics within each group respect the expected learnability ordering: mean $s(x)$ decreases monotonically from STABLE LEARNER (0.521) through SLUGGISH LEARNER (0.470) and INEFFECTIVE LEARNER (0.391), while the fraction of tasks collapsed to the probability floor rises correspondingly, reaching

92% for STUBBORNLY UNLEARNED. Third, distributions under $\mathcal{P}_1$ and $\mathcal{P}_2$ are nearly identical across all profiles, confirming that the score reflects a stable property of the data rather than an artifact of any particular probe realization.

		$\mathcal{P}_1$			$\mathcal{P}_2$		
Profile	n	Share	Mean	Floor%	Share	Mean	Floor%
STABLE LEARNER	200	18.0%	0.521	30.5%	16.3%	0.456	39.0%
SLUGGISH LEARNER	394	32.0%	0.470	30.5%	31.9%	0.454	32.0%
INEFFECTIVE LEARNER	253	17.1%	0.391	31.2%	17.4%	0.386	31.6%
ALREADY MASTERED	659	20.7%	0.182	63.3%	21.4%	0.182	63.6%
STUBBORNLY UNLEARNED	527	11.7%	0.129	92.4%	12.1%	0.129	93.6%
FORGETTING	15	0.5%	0.210	73.3%	0.9%	0.339	53.3%
LEARNABLE (combined)	847	67.0%	0.459	30.7%	65.6%	0.434	33.5%
NON-LEARNABLE (combined)	1201	33.0%	0.159	76.2%	34.4%	0.160	76.6%

Table 6: Per-profile distribution of TRAJVAL scores under two independent probe sets. **Share**: fraction of total sampling weight. **Mean**: mean $s(x)$ within profile. **Floor%**: fraction of tasks at the probability floor.

Score stability under varying probe horizons. TRAJVAL derives its score from θ_{probe} obtained after T epochs of probe training. We examine how early the resulting score ordering stabilizes by saving intermediate checkpoints $\theta^{(k)}_{\text{probe}}$ at each epoch $k \in \{1, \ldots, T\}$ ($T = 20$), recomputing $\text{post}_k(x)$ over the full pool $\mathcal{D}$, and measuring the Spearman rank correlation ρ between the intermediate score $s_k(x) = (1 - \text{pre}(x)) \cdot \max(0, \text{post}_k(x) - \text{pre}(x))$ and the final score $s_T(x)$.

As shown in Figure 9, ρ rises rapidly in the early epochs and reaches 0.85 by $k = 14$, well before the nominal probe horizon $T = 20$. Beyond $k = 14$, the correlation remains stable and the two independently drawn probe sets stay within 0.04 of each other at all epochs. This indicates that the relative learnability ordering captured by TRAJVAL crystallizes early in probe training, and that the probe budget could be reduced to roughly 70% of the nominal horizon without meaningfully degrading score quality.

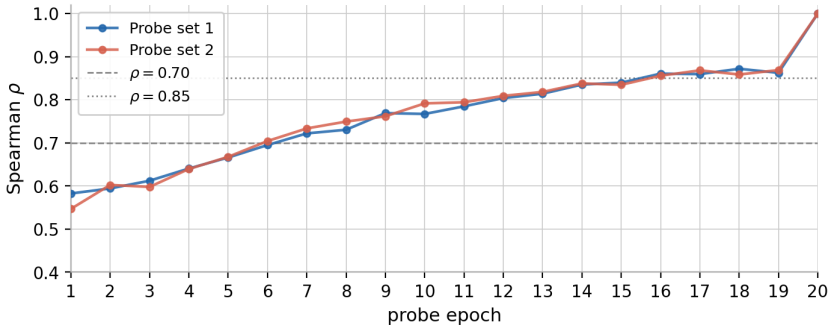


Figure 9: Spearman rank correlation between $s_k(x)$ and $s_T(x)$ as a function of probe epoch k, evaluated on $\mathcal{D}$. Two independently sampled probe sets are shown. The ordering stabilizes ($\rho \geq 0.85$) by $k = 14$, well before the full probe horizon $T = 20$.

B.5 COMPUTATIONAL COST OF TRAJVAL

Cost Structure. The preprocessing overhead of TRAJVAL decomposes into two components. The dominant cost is the probe training run, in which the base model θ_0 is fine-tuned via RL on a uniformly sampled probe set $\mathcal{P} \subset \mathcal{D}$. In our primary configuration, $|\mathcal{P}| = 512$ for a pool of $|\mathcal{D}| = 17{,}398$ tasks, corresponding to approximately 3% of the full training pool; the total probe training cost is therefore $O(T \cdot |\mathcal{P}|)$. The secondary cost is endpoint estimation: for every task $x \in \mathcal{D}$, we evaluate $\bar{r}_{\text{early}}(x)$ and $\bar{r}_{\text{late}}(x)$ via two targeted inference passes over the full pool using the recorded probe

checkpoints, incurring $O(2 \cdot |\mathcal{D}|)$ forward passes with no gradient computation. Crucially, once the per-task sampling weights $\{p(x)\}$ are computed, TRAJVAL introduces *no additional overhead during the main training phase*: the weights are fixed prior to training, and sampling at each step reduces to a single weighted multinomial draw.

Empirical Overhead Measurement. Total training time is dominated by model rollout length—a factor orthogonal to our method—we focus on the *experience pipeline processing time* (`time/experience_pipeline/total`) as the primary measurement target. This metric captures the end-to-end latency of the data-processing pipeline at each training step, encompassing experience deserialization, advantage computation, and output buffer writes, and is the only pipeline-level signal that can reflect task-selector overhead in our logging infrastructure. It explicitly excludes model inference and parameter update time, making it a clean proxy for the marginal cost.

Statistic	Random	TRAJVAL	Abs. Diff.	Change
Mean	2.301 s	2.247 s	−0.055 s	−2.4%
Median	2.286 s	2.304 s	+0.018 s	+0.8%
p90	2.957 s	2.971 s	+0.014 s	+0.5%
p95	3.177 s	3.173 s	−0.003 s	−0.1%
Cumulative (271 steps)	623.6 s	608.9 s	−14.8 s	−2.4%

(a) Experience pipeline total time (`time/experience_pipeline/total`).

Statistic	Random	TRAJVAL	Abs. Diff.	Pipeline Share
Mean latency	0.019 ms	0.239 ms	+0.220 ms	0.010%
p99 latency	0.033 ms	0.253 ms	+0.221 ms	0.011%
Cumulative (271 steps)	5.2 ms	64.8 ms	+59.4 ms	0.010%

(b) Micro-benchmark: `get_indices()` latency over 1,000 independent calls; pipeline share computed relative to the 623.6 s random baseline.

Table 7: Empirical overhead of TRAJVAL relative to random sampling. **Top**: Experience pipeline processing time over 271 training steps. Mean and percentile differences are within measurement noise. **Bottom**: Micro-benchmark isolating `operator` latency over 1,000 calls. Although the relative increase appears large, the absolute per-step cost is equivalent to 0.01% of total pipeline time.

Table 7 (top) reports the pipeline-level measurements over 271 training steps. Across the reported percentiles up to p95, the difference between TRAJVAL and random sampling remains below 0.02 s, well within the variability attributable to system-level noise such as network I/O and serialization. The cumulative pipeline times are statistically indistinguishable. To further isolate the selector's contribution, Table 7 (bottom) presents a direct micro-benchmark of the `Trinity-Operator` call. TRAJVAL's weighted multinomial sampling takes approximately 0.24 ms per call, compared to 0.02 ms for uniform random sampling. While the relative difference is large in percentage terms, the absolute per-step overhead is 0.22 ms—accumulating to roughly 59 ms over 271 steps, or 0.01% of the total pipeline time. This confirms that the learnability-aware sampling logic is entirely subsumed by other pipeline components and does not constitute a measurable bottleneck in practice.

Cost-Benefit Summary. The marginal cost of TRAJVAL—a one-time probe run on $\approx 3\%$ of the training pool, two inference passes for endpoint estimation, and a negligible per-step selector overhead—is one-time and can be amortized across runs (Appendix B.6). As reported in Section 5.2, TRAJVAL-augmented methods consistently reduce the Steps-to-Baseline (S2B) metric in most configurations, with S2B as low as 40.0% on Logic at the 4B scale. This indicates that the same baseline peak accuracy is achievable in less than half the training steps.

B.6 CROSS-REGIME REUSE OF THE LEARNABILITY PRIOR

Since learnability is regime-conditional, estimating the prior on the target regime itself (*in-regime*) is our default and most accurate setting. As re-running the probe for every regime repeats the one-time preprocessing cost of Appendix B.5, we ask whether an existing prior can be reused: (i) *same-family,*

Target	Prior source	Best Acc↑	AUC↑	S2B↓
Qwen3-4B				
Random	–	0.5060	0.4829	–
+ TRAJVAL (transfer)	Qwen3-1.7B	0.5180	0.4889	66.0
+ TRAJVAL (transfer)	Llama-3.2-3B	0.5129	0.4817	84.0
+ TRAJVAL (*in-regime*)	Qwen3-4B	*0.5190*	*0.4931*	*60.0*
Qwen3-8B				
Random	–	0.5342	0.4999	–
+ TRAJVAL (transfer)	Qwen3-4B	0.5484	0.5109	87.5

Table 8: Reusing the TRAJVAL prior across regimes (Math). Each transferred prior is estimated on the *prior source* and guides RL on the *target* without re-estimation. S2B is the fraction of steps to reach the Random baseline's peak accuracy (lower is better; "–" marks that baseline). The in-regime row is a reference, not a competing method.

cross-scale, from a smaller to a larger model in the same family; and (ii) *cross-family*, from a different family. All runs are on Math; each transferred prior guides RL on the target without re-estimation, against uniform sampling (Random / BASE-GRPO) on that target, with the in-regime result on Qwen3-4B as a reference.

Same-family, cross-scale reuse. Within Qwen3, a prior from a smaller model remains useful on a larger one, improving all three metrics over uniform sampling: Qwen3-1.7B→Qwen3-4B reaches the baseline peak in 66.0% of steps, and Qwen3-4B→Qwen3-8B in 87.5% (Table 8). A single prior can thus be shared across scales rather than recomputed per model.

Cross-family reuse. Transfer across families also helps, but less. Llama-3.2-3B→Qwen3-4B raises Best Acc (0.5060→0.5129) and reaches the peak in 84.0% of steps, while AUC is essentially unchanged (0.4817 vs. 0.4829). Both the smaller Best Acc gain and the flat AUC are consistent with a prior less matched to the target regime.

Overall, the in-regime prior on Qwen3-4B is strongest (Best Acc 0.5190), slightly above same-family (0.5180) and cross-family (0.5129) transfer; same-family reuse stays closest to it and converges fastest (S2B 66.0 vs. 84.0). In-regime estimation therefore remains the default for peak performance, while transfer is a lower-cost alternative that spreads the probe cost across models—same-family reuse being the more dependable of the two.

	MATH500			Minerva			Olympiad			AMC23			AIME25			AIME24		
Method	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B
Random	75.5	72.1	–	19.6	18.0	–	43.2	39.9	–	60.4	53.1	–	17.6	15.3	–	21.9	18.7	–
+Ours	**76.3**	**72.9**	**75.8**	19.5	**18.4**	–	44.0	**40.5**	**84.4**	**62.6**	**55.1**	**90.9**	**20.6**	**16.5**	**45.5**	**24.0**	**20.3**	**50.5**
BOTS	75.5	72.1	–	19.7	18.1	–	44.1	39.9	–	61.1	54.1	–	18.8	15.7	–	23.5	18.9	–
+Ours	**76.6**	**72.9**	**94.4**	**20.0**	18.1	**95.0**	**44.9**	**40.9**	**85.0**	**62.4**	**54.9**	**95.0**	18.5	**15.9**	–	23.8	**19.5**	**75.0**
GRESO	75.3	71.7	–	20.1	18.4	–	44.1	39.7	–	60.9	53.4	–	18.2	15.4	–	23.1	18.8	–
+Ours	**76.0**	**72.6**	**85.0**	19.7	18.3	–	**44.9**	**40.2**	**90.0**	**61.8**	**54.3**	**100.0**	**19.0**	**15.9**	**84.2**	**24.3**	**19.6**	**85.0**

Table 9: Per-benchmark results on mathematical reasoning, Qwen3-1.7B. Best Acc (↑), AUC (↑), and S2B (%, ↓) for each benchmark and method. Best results per column in **bold**.

B.7 HYPER-PARAMETER SENSITIVITY: POWER COMPRESSION EXPONENT

Table 11 reports the sensitivity of TRAJVAL to the power compression exponent α, which controls the degree to which learnability scores are flattened before conversion to sampling probabilities. Across three values $\alpha \in \{0.1, 0.3, 0.5\}$, the overall pattern of improvement over the RANDOM baseline is preserved on both domains. On Math, all three settings advance Best Acc beyond the random baseline (0.3967), and $\alpha = 0.3$ achieves the most favorable convergence speed, reaching the random baseline's peak accuracy in only 65.7% of the steps while maintaining a competitive AUC of 0.3730. On Logic, the Best Acc increases monotonically with α from 0.1884 to 0.1924, while $\alpha = 0.3$ yields

	MATH500			Minerva			Olympiad			AMC23			AIME25			AIME24		
Method	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B	Best	AUC	S2B
Random	82.6	80.9	–	24.8	23.5	–	52.2	50.0	–	79.5	75.8	–	28.4	27.0	–	36.3	32.6	–
+Ours	**83.8**	**81.9**	**60.0**	24.6	**23.5**	–	**53.8**	**51.0**	**60.0**	**80.9**	**77.8**	**60.0**	**31.3**	**28.6**	**43.6**	**37.6**	**32.9**	**72.0**
BOTS	82.9	80.9	–	24.4	23.4	–	52.0	49.3	–	80.5	75.0	–	29.5	26.2	–	35.9	31.2	–
+Ours	**83.8**	**81.5**	**70.0**	**24.8**	**23.5**	**94.4**	**53.5**	**50.4**	**68.4**	**80.5**	**76.3**	**95.0**	**31.0**	**27.9**	**65.0**	**37.4**	**32.2**	**75.0**
GRESO	83.2	80.8	–	**24.9**	23.3	–	51.6	48.5	–	**81.3**	74.2	–	29.0	25.4	–	35.4	29.6	–
+Ours	**83.5**	**81.2**	**80.0**	25.1	**23.4**	**100.0**	**53.3**	**49.9**	**73.7**	80.3	**74.6**	–	**31.2**	**27.1**	**70.0**	**37.2**	**31.4**	**75.0**

Table 10: Per-benchmark results on mathematical reasoning, Qwen3-4B. Best Acc (↑), AUC (↑), and S2B (%, ↓) for each benchmark and method. Best results per column in **bold**.

the highest AUC (0.1663) among all variants, suggesting that a moderate compression level best balances priority concentration and distributional coverage.

Taken together, these results indicate that TRAJVAL is not sensitive to the precise choice of α within the range evaluated: all three settings produce non-trivial improvements over uniform sampling on both domains and both metrics, and no single value of α degrades performance below the RANDOM baseline. The moderate value $\alpha = 0.3$ provides the best overall trade-off and is therefore adopted as the default throughout the main experiments.

	Math (Avg)			Logic		
Method	S2B (%) ↓	Best Acc ↑	AUC ↑	S2B (%) ↓	Best Acc ↑	AUC ↑
Random	–	0.3967	0.3620	–	0.1826	0.1544
TrajVal ($\alpha = 0.1$)	78.79	0.4062	0.3656	69.57	0.1884	0.1611
TrajVal ($\alpha = 0.3$)	65.66	0.4105	0.3730	73.91	0.1904	0.1663
TrajVal ($\alpha = 0.5$)	78.79	0.4071	0.3647	69.57	0.1924	0.1660

Table 11: Sensitivity to the power compression exponent α on Qwen3-1.7B. All other hyperparameters are held fixed.

B.8 TEMPORAL VALIDITY OF THE STATIC LEARNABILITY PRIOR

A defining choice of TRAJVAL is that the learnability prior is estimated once at the base model θ_0 from a short probe run and then kept fixed throughout main training. Our main experiments show that this one-shot prior improves data efficiency across models, training strategies, and evaluation benchmarks. Since the prior is static while the policy keeps evolving, this section takes a closer look at how the learnability signal it encodes at θ_0 evolves as training progresses, providing a mechanistic view of why a one-shot estimate remains useful over a long training horizon.

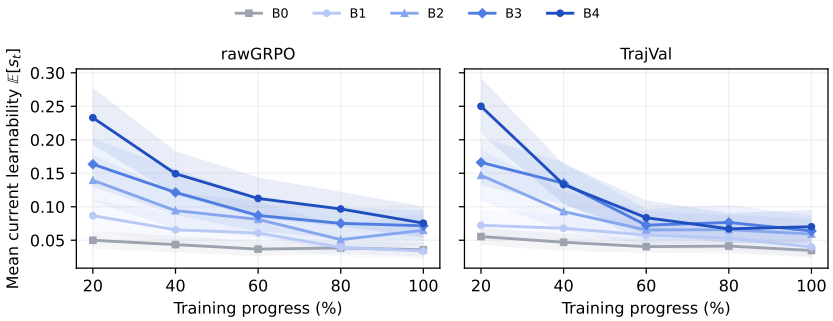


Figure 10: Mean current learnability $\mathbb{E}[s_t]$ of fixed initial-score buckets over training progress. B_0 contains tasks with $s_0 = 0$; B_1–B_4 are quartile groups of the positive initial scores in increasing order. Shaded regions denote 95% bootstrap CIs over tasks. Left: RAWGRPO (uniform sampling). Right: TRAJVAL (prior-weighted sampling).

Since TRAJVAL consumes the prior through probabilistic sampling, tasks with similar scores need not retain their exact order for the prior to remain useful, so we analyze it at the group level. At θ_0 we place tasks with $s_0(x) = 0$ into B_0 and split those with $s_0(x) > 0$ into quartile buckets B_1–B_4; membership is fixed thereafter. To probe how long the initial ranking remains informative, we treat each checkpoint θ_t of the main training run as a fresh base model and repeat the same probe-training protocol used to estimate s_0: starting from θ_t, we run probe training on the same probe set P under identical hyperparameters, and evaluate every x in a shared audit pool of 1,024 tasks before and after probing to obtain $\mathrm{pre}_t(x)$ and $\mathrm{post}_t(x)$; the current learnability $s_t(x) = (1 - \mathrm{pre}_t(x)) \max(0, \mathrm{post}_t(x) - \mathrm{pre}_t(x))$ is then computed with the same formula as s_0. We repeat this procedure at five checkpoints on two Qwen3-1.7B Math runs, RAWGRPO and TRAJVAL, over which policy drift $d_t = \mathbb{E}_x|\mathrm{pre}_t(x) - \mathrm{pre}_0(x)|$ grows from 0.14 to 0.22.

Group-level separation persists under policy drift. The buckets separate early and retain their order throughout: at the end of training, B_2–B_4 remain above B_0 and B_1, with the top bucket roughly twice as learnable as the zero-score bucket (0.076 vs. 0.036 for RAWGRPO, 0.070 vs. 0.035 for TRAJVAL). The per-task Spearman correlation $\rho(s_0, s_t)$ decreases from 0.39 to 0.20 yet stays significantly positive at every checkpoint ($p < 1.2 \times 10^{-7}$), so the initial ranking continues to carry information about the current one throughout training.

The overall decline reflects consumed headroom, not a failed prior. All buckets shift downward as the policy improves and fewer tasks remain improvable within a short probe; B_4 drops from 0.23 to 0.08. The two runs age almost identically, so the decay is a general property of RL post-training rather than an artifact of prior-weighted sampling. Consistent with the intended mechanism, top buckets decline faster and flatten earlier under TRAJVAL, which samples high-value tasks more often and exhausts their headroom sooner; this mirrors the faster convergence of TRAJVAL reported in the main results.

Learnability is thus a distributional signal that evolves during training: a one-shot estimate remains effective over a long horizon, and modeling its evolution through adaptive priors is a natural direction for future work.

C COMPLETE EXPERIMENTAL RESULTS

C.1 ROLLOUT DYNAMICS UNDER LEARNABILITY-AWARE SAMPLING

Figure 11 visualizes the training-time rollout accuracy under different sampling strategies for Qwen3-1.7B on Math and Logic. These curves are intended as a diagnostic view of how TRAJVAL reshapes the sampled training stream. They are ***not*** a direct substitute for downstream benchmark performance, which is reported in Section 5.2.

- ***Earlier rise in rollout accuracy.*** Across both domains, the TRAJVAL-augmented variants generally increase rollout accuracy faster than their corresponding baselines, with a clearer separation on Math. This pattern is consistent with the intended effect of learnability-aware sampling: allocating more updates to tasks whose rewards are more responsive to continued optimization.
- ***Baseline-specific behavior is preserved early on.*** In the early stage of training, each augmented curve remains close to its underlying baseline, rather than collapsing to a common trajectory. This suggests that TRAJVAL acts as a mild prior on top of the original scheduler, instead of overriding its native sampling behavior.
- ***Greater alignment later in training.*** As training proceeds, the augmented variants become more similar to one another than the original baselines do, especially on Math. We view this as qualitative evidence that different schedulers equipped with the same learnability prior are gradually steered toward a more similar set of high-potential samples.

The contrast between Math and Logic is also informative: the same qualitative trends appear in both domains, but the Logic curves are noisier and the separations are smaller. This may reflect greater heterogeneity in the logic training pool, making rollout-level improvements less smooth even when the sampling prior is beneficial.

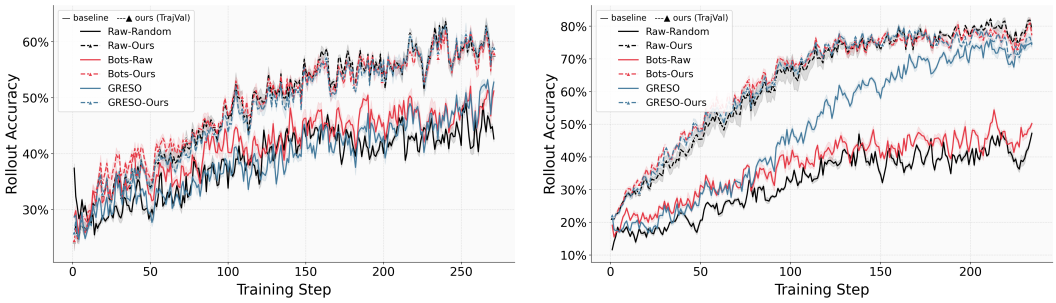


Figure 11: Training-time rollout accuracy under different sampling strategies for Qwen3-1.7B. Left: Math. Right: Logic. Solid lines denote the original baselines, and dashed lines denote their TRAJVAL-augmented counterparts. The plots provide a qualitative view of how learnability-aware sampling changes the training dynamics on the sampled pool.

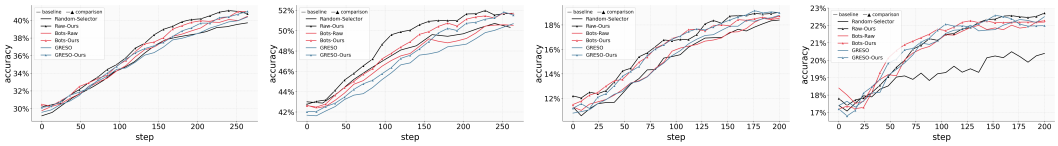


Figure 12: Evaluation accuracy curves for all main configurations on Math and Logic. The x-axis shows training steps and the y-axis shows evaluation accuracy.

C.2 EVALUATION CURVES

Figure 12 presents the full evaluation trajectories for the main configurations across Math and Logic. Several consistent patterns are visible. First, TRAJVAL-augmented variants generally rise earlier than their corresponding baselines, indicating improved data efficiency in the early and middle stages of training. Second, these gains are not limited to faster initial progress: in most settings, the augmented curves also maintain a higher or comparable plateau, consistent with the improvements in Best Acc and AUC reported in Table 1. Third, the qualitative trend is shared across both standalone and plug-in use cases, suggesting that the learnability prior remains beneficial whether used on its own or combined with an online scheduler. Overall, the trajectory-level view reinforces the main conclusion that learnability-aware allocation improves training efficiency without sacrificing final evaluation performance.

C.3 COMPLETE PER-BENCHMARK RESULTS

Tables 9 and 10 report per-benchmark breakdowns of the main Math results for Qwen3-1.7B and Qwen3-4B, respectively. Each table reports Best Acc, AUC, and S2B for all six evaluation benchmarks (MATH500, Minerva, OlympiadBench, AMC23, AIME25, AIME24) under all three base training strategies and their TRAJVAL-augmented counterparts. Results on Llama-3.2-3B are reported separately in Appendix B.3.

D ALGORITHM AND PSEUDOCODE OF TRAJVAL

Algorithm 1 summarizes the full procedure.

Algorithm 1 TRAJVAL: Probe-Based Learnability Estimation for RL Post-Training

Input: training pool $\mathcal{D}$, base model θ_0, probe size $|\mathcal{P}|$, probe epochs T, checkpoint window size K, compression exponent α, floor ϵ
Output: sampling distribution $\{p(x)\}_{x\in\mathcal{D}}$

// Phase I: estimate a learnability prior from a probe run
Sample a probe set $\mathcal{P} \subset \mathcal{D}$ uniformly, with $|\mathcal{P}| \ll |\mathcal{D}|$
Train θ_0 on $\mathcal{P}$ for T epochs to obtain probe checkpoints $\{\theta_t\}$
Select early checkpoints $\mathcal{T}_{\text{early}}$ and late checkpoints $\mathcal{T}_{\text{late}}$, where $|\mathcal{T}_{\text{early}}| = |\mathcal{T}_{\text{late}}| = K$
for each task $x \in \mathcal{D}$ **do**
    Compute endpoint estimates

$$\bar{r}_{\text{early}}(x) = \frac{1}{K} \sum_{t\in\mathcal{T}_{\text{early}}} \text{acc}(x, \theta_t), \qquad \bar{r}_{\text{late}}(x) = \frac{1}{K} \sum_{t\in\mathcal{T}_{\text{late}}} \text{acc}(x, \theta_t)$$

end for
$\{p(x)\}_{x\in\mathcal{D}} \leftarrow \textsc{ComputeWeights}\big(\{\bar{r}_{\text{early}}(x), \bar{r}_{\text{late}}(x)\}_{x\in\mathcal{D}}, \alpha, \epsilon\big)$

// Phase II: main RL training with prior-weighted sampling
for training step $t = 1, 2, \ldots$ **do**
    Sample a batch from $\mathcal{D}$ according to $p(x)$
    or use the composed weight $w_{\text{final}}(x, t) = p(x)\, w_{\text{online}}(x, t)$
    Perform one standard RL update on the sampled batch
end for

function $\textsc{ComputeWeights}(\{\bar{r}_{\text{early}}(x), \bar{r}_{\text{late}}(x)\}, \alpha, \epsilon)$
    for each task $x \in \mathcal{D}$ **do**
        $\Delta(x) \leftarrow \bar{r}_{\text{late}}(x) - \bar{r}_{\text{early}}(x)$
        $s(x) \leftarrow \big(1 - \bar{r}_{\text{early}}(x)\big) \max\big(0, \Delta(x)\big)$
        $w(x) \leftarrow \max\big(s(x)^{\alpha}, \epsilon\big)$
    end for
    return $p(x) = \dfrac{w(x)}{\sum_{x'\in\mathcal{D}} w(x')}$ for all $x \in \mathcal{D}$
end function