Sample Learning State Guided Dynamic SFT-RL Integration Post-training

Anonymous ACL submission

Abstract

The joint optimization of Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) has emerged as a prominent paradigm for LLM post-training. Current methods, however, either sequence these stages or combine them with static weights, thereby overlooking sample differences and model dynamics, which can lead to overfitting or reward hacking. To address this, we introduce Sample Learning State (SLS) characterized by two key metrics: the Degree of Sample Mastery and the Dispersion of Exploration Trajectory, which capture the dynamic features in the model's handling of samples during the training process. Based on SLS, we design a training-state-aware, sample-wise weighting coefficient that enables dynamic integration of SFT and RL loss, balancing supervised guidance and autonomous exploration for synergistic optimization. Extensive experiments demonstrate that our method achieves new state-of-the-art (SOTA) results on four in-domain mathematical benchmarks and two out-of-domain tasks. Moreover, it exhibits strong robustness in multi-reward scenarios, effectively mitigating reward hacking under auxiliary constraints while maintaining stable reasoning performance. We will release the code upon publication.

1 Introduction

The remarkable capabilities of Large Language Models (LLMs) are primarily attributed to the two-stage training paradigm: pre-training on massive corpora to build foundational knowledge (Vaswani et al., 2017; Wei et al., 2022; Touvron et al., 2023a), and post-training to elicit specific capabilities and align with human values (Stiennon et al., 2020; Ouyang et al., 2022). During post-training, Supervised Fine-Tuning (SFT) (Wang et al., 2023; Chung et al., 2024; Zhang et al., 2025b) leverages high-quality instruction data to activate latent capabilities, whereas Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Schulman et al., 2017) aligns model behavior with human preferences. More recently, for tasks with objectively verifiable criteria, such as mathematical reasoning, Reinforcement Learning with Verifiable Rewards (RLVR) (Zhang et al., 2025a; Shao et al., 2024) has emerged as the predominant approach.

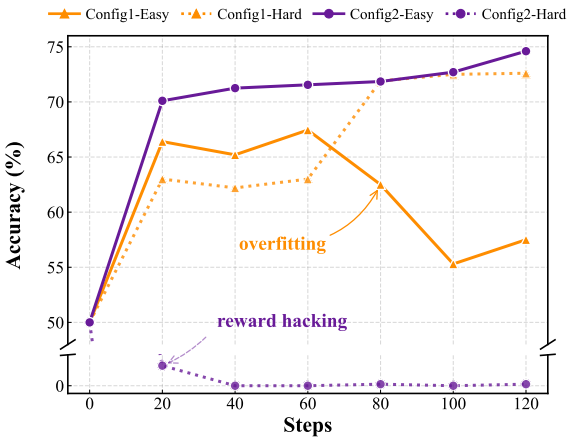

Figure 1: The static weighting scheme for SFT-RL integration lacks adaptability to different sample types.

Both SFT and RL have inherent limitations(Chen et al., 2025a; Chu et al., 2025). SFT is provided with explicit supervision signals, its excessive reliance on imitating reference trajectories confines the model to surface-level pattern replication, resulting in spurious reasoning traces that lack logical grounding (Wei et al., 2021; Zhou et al., 2023; Wu et al., 2025). In contrast, RL optimizes reasoning by exploring diverse solution paths, yet when processing complex samples beyond the model's current capability boundary, the absence of effective supervisory guidance makes the model susceptible to reward hacking and other unintended behaviors (Skalse et al., 2022; Gao et al., 2023). Consequently, integrating SFT and RL has become the mainstream strategy for LLM post-training. Early non-reasoning LLMs(Ouyang et al., 2022)adopted a sequential

"SFT-RL" pipeline, whereas reasoning models such as DeepSeek-R1 employ an interleaved "SFT-RL-SFT-RL" scheme (DeepSeek-AI et al., 2025).

Although these fusion strategies have demonstrated notable improvements, they fundamentally treat SFT and RL as independent training stages, causing the performance ceiling of the RL phase to be constrained by the initial model's capabilities (Yue et al., 2025; Nguyen et al., 2025), thereby impeding policy optimization. To address this limitation, recent work has explored deeper integration of SFT and RL. For instance, LUFFY (Yan et al., 2025) guides reasoning by jointly leveraging on-policy and off-policy trajectories; KDRL (Xu et al., 2025) converts supervision signals into regularization terms for joint training with GRPO; and CHORD (Zhang et al., 2025c) incorporates SFT supervision into RL training via globally predefined weights. Nevertheless, these methods rely on static fusion mechanisms, applying globally predefined weighting coefficients that remain invariant regardless of model proficiency or sample difficulty.

We find that predefined static weighting fundamentally limits a deeper synergy between SFT and RL. Since different types of samples exhibit significantly distinct demands for SFT and RL, a fixed weighting scheme cannot simultaneously tailor the optimal learning strategy for diverse samples, as shown in Figure 1. Where, we categorize the samples in OpenR1-Math dataset into three groups based on their performances on qwen2.5-math-7b: easy samples (at least 3 correct out of 4 rollouts), hard samples (at most 1 correct), and others. We examine two representative static weighting configurations. Config1 employs high SFT weights and low RL weights, while Config2 adopts the opposite one. As shown in Figure 1, under Config1, hard samples benefit from strong supervision and learn effectively (Config1-Hard), whereas easy samples improve rapidly during early training (first 20 steps) but subsequently overfit and degrade continuously (Config1-Easy). Config2 enables steady improvement on easy samples (Config2-Easy), but insufficient supervision causes hard samples to shift optimization focus toward secondary rewards (e.g., formatting compliance), triggering reward hacking and rapid degradation on the core task (Config2-Hard). These results indicate that predefined fixed fusion weights cannot adapt to varying sample characteristics, failing to leverage the respective strengths of different learning algorithms.

To address the problem, we propose a way to dynamic weighting of SFT-RL integration based on sample learning state to achieve optimal synergy between the two learning methods. To this end, we first introduce the Sample Learning State (SLS), which defines dynamic metrics for the model's handling of samples during the training process. SLS not only varies across samples but also evolves with the model's training process. Specifically, SLS is a tuple, where the first element is the degree of mastery of the model over a sample under consideration, and the second element captures the dispersion of exploration trajectories on that sample. We then design a training-state-aware, sample-wise weighting coefficient as a nonlinear function of the two elements, enabling dynamic integration of the SFT and RL losses. The mechanism also excels in multi-reward settings, where dynamic weighting avoids the common pitfall of core performance degradation caused by over-optimizing secondary rewards (e.g., formatting), a frequent issue with pure RL or static fusion method.

Extensive experiments validate the effectiveness of our method. It achieves new State-Of-The-Art (SOTA) across four in-distribution (ID) mathematical reasoning benchmarks and two out-of-distribution (OOD) tasks on both Qwen2.5-Math-1.5B and 7B models, significantly outperforms all static weighting baselines. Experiments under multi-reward settings further demonstrate that our method optimizes auxiliary objectives while maintaining stable improvement on core task performance when secondary preference rewards are introduced. In summary, our main contributions are as follows:

- We propose the Sample Learning State (SLS) including the degree of sample mastery and the dispersion of exploration trajectory, which characterizes the dynamic variations in the model's handling of samples during the training process.
- Based on SLS, we design a training-state-aware, sample-wise weighting coefficient α_i that enables dynamic integration of SFT and RL loss, balancing supervised guidance and autonomous exploration for synergistic optimization.
- Our method achieves new SOTAs across ID mathematical reasoning and OOD benchmarks on both Qwen2.5-Math-1.5B and 7B models. It also optimizes auxiliary objectives without compromising core reasoning performance when secondary preference rewards are introduced.

2 Related Work

Existing post-training approaches (Achiam et al., 2023; Touvron et al., 2023b; DeepSeek-AI et al., 2025) adopt a sequential "SFT warm-up followed by RL optimization" pipeline. They are prone to deviating from core objectives during exploration due to the lack of continuous supervision.

Subsequent research have developed three main approaches for deeper integration. The first is *Alternating update* method (e.g., ReLIFT (Ma et al., 2025), SASR (Chen et al., 2025b)), which achieve synergy through interleaved RL policy updates and SFT supervision steps. The second is *Trajectory-guided* method (e.g., LUFFY (Yan et al., 2025), SRFT (Fu et al., 2025),TRAPO (Su et al., 2025)), which leverages offline expert demonstrations to constrain RL exploration. The third is *Loss weighting* method (e.g., KDRL (Xu et al., 2025), RPO (Liu et al., 2024), CHORD (Zhang et al., 2025c)), which incorporates SFT-related penalty terms to regularize policy drift.

Recent work shows that SFT and RL can be unified within the same gradient optimization framework, differing in data sources and gradient weighting (Lv et al., 2025; Wu et al., 2025), providing a theoretical foundation for fine-grained integration.

3 Method

3.1 Sample Learning State

We denote the initial model parameters as θ_0. After t training steps, the parameters are updated to θ_t, which correspond to the model m_{θ_t} and its current policy π_{θ_t}. Let $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{N}$ be N samples for post-training.

For each sample s_i, we generate an exploration trajectory set $\mathcal{O}_i = \{o_{i,j}\}_{j=1}^{G}$ by performing G independent rollouts on the current model m_{θ_t}. Each trajectory $o_{i,j}$ is associated with a core task reward $r_{i,j}^{\text{core}} \in \{0, 1\}$ and a set of K auxiliary preference rewards $\{r_{i,j}^{k}\}_{k=1}^{K}$ (e.g., format compliance).

Based on these definitions, we formulate the **Sample Learning State** (SLS) for sample s_i at training step t as a tuple:

$$\text{SLS}_i(t) = \langle M_i(t), D_i(t) \rangle \tag{1}$$

where $M_i(t)$ quantifies the model's proficiency on s_i, and $D_i(t)$ characterizes the dispersion across exploration trajectories. Together, these components govern the adaptive allocation of SFT and RL weights for sample s_i at the current training stage.

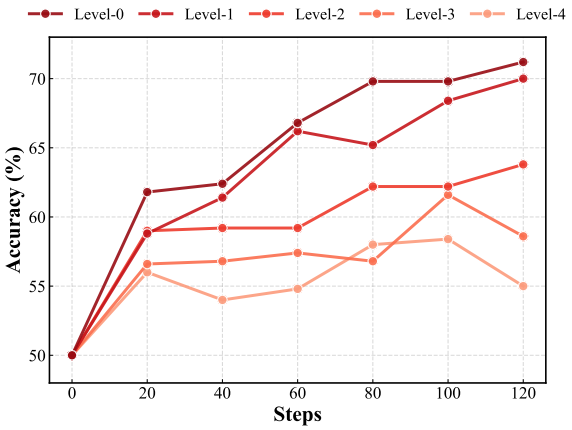

Figure 2: SFT performance across sample mastery (M) levels. **Darker** colors denote **lower** mastery.

The Degree of Sample Mastery $M_i(t)$ The mastery score $M_i(t)$ measures the reliability of model m_{θ_t} in completing the core task on sample s_i, defined as the average core task reward across G sampled trajectories:

$$M_i(t) = \frac{1}{G} \sum_{j=1}^{G} r_{i,j}^{\text{core}} \tag{2}$$

This metric is intrinsically coupled with the efficacy of SFT: for low-mastery samples (low $M_i(t)$), external supervision effectively rectifies the model's performance deficiencies. Conversely, for samples that already exhibit consistent task proficiency (high $M_i(t)$), excessive supervision may trigger surface-level overfitting or interfere with the model's intrinsic reasoning trajectories.

To verify this correlation, we divide the OpenR1-Math and GSM8K datasets into five mastery levels (Level 0–4) based on $M \in [0, 1]$. We reserve 100 random instances from each level for validation and utilize the remainder for SFT training.

As illustrated in Figure 2, the performance gains induced by SFT exhibit a significant negative correlation with sample mastery. Specifically, supervision drives consistent improvements on low-mastery samples (Level 0), whereas imposing redundant supervision on high-mastery samples (Level 4) yields diminishing returns or even performance regression.

These findings suggest that SFT is paramount for bridging capability gaps in unmastered samples. Conversely, imposing redundant supervision on already-mastered instances risks including surface-level overfitting and disrupting intrinsic reasoning trajectories, thereby undermining generalization.

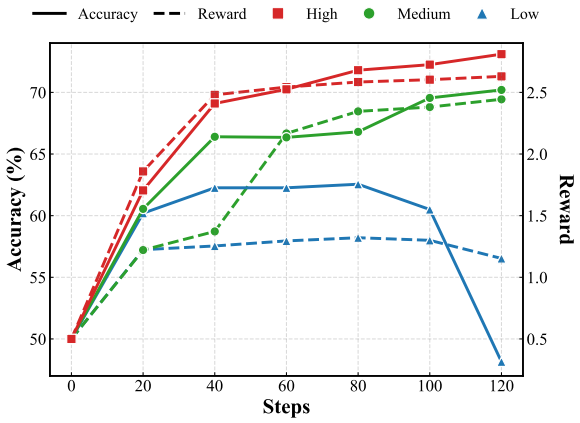

Figure 3: RL performance across varying dispersion levels (D). **Accuracy** and **Reward** denote the core task accuracy and the total reward, respectively.

The Dispersion of Exploration Trajectories $D_i(t)$ The dispersion metric $D_i(t)$ measures the variability of reasoning trajectories generated by model m_{θ_t} on sample s_i, defined as the standard deviation of the total rewards across G sampled trajectories:

$$D_i(t) = \text{Std}\left(\{R_{i,j}\}_{j=1}^{G}\right) \tag{3}$$

where $R_{i,j} = r_{i,j}^{\text{core}} + \sum_{k=1}^{K} r_{i,j}^{k}$ denotes the composite reward for the j-th trajectory, aggregating the core task alignment and K auxiliary preference scores.

This metric reflects the potential of exploration trajectories to yield effective gradient signals for policy optimization, and is directly tied to RL training efficacy. When $D_i(t)$ is high, significant reward variance provides highly discriminative contrastive signals, facilitating the rapid distinction between superior and inferior reasoning paths. Conversely, when $D_i(t)$ is low, trajectory homogeneity leads to ineffective optimization guidance, making the model prone to reward hacking by over-optimizing auxiliary rewards (e.g., format compliance) at the expense of core task objectives.

To validate this correlation, we construct a composite reward function comprising a core correctness reward $r_{i,j}^{\text{core}} \in \{0, 1\}$ and an auxiliary format reward $r_{i,j}^{\text{format}} \in [0, 2.1]$. We partition samples from both datasets into three distinct groups based on equal-spaced intervals of the dispersion metric $D_i(t)$ (Low, Medium, and High), and conduct RL training independently for each group.

As illustrated in Figure 3, the high-dispersion group exhibits faster reward convergence and significant gains in core task accuracy, whereas the low-dispersion group achieves only marginal reward growth, often accompanied by regression in core task performance. These findings suggest that when exploration trajectories lack sufficient diversity, relative-reward-based policy updates struggle to obtain effective gradient signals, and may instead induce the model to optimize secondary objectives, thereby triggering reward hacking.

These experiments conclusively validate the significant impact of sample heterogeneity on SFT and RL training effectiveness, providing a core rationale for sample-level dynamic fusion: the SFT-RL weight allocation must be adaptively calibrated based on each sample's real-time learning state.

3.2 SLS-Guided Dynamic SFT-RL Fusion

Building upon the Sample Learning State $\text{SLS}_i(t) = \langle M_i(t), D_i(t) \rangle$ defined in Section 3.1, we establish a single-stage, sample-level adaptive SFT-RL joint optimization framework. At training step t, for each sample s_i, we dynamically calibrate a fusion weight $\alpha_i(t) \in [0, 1]$ (omitting t for brevity below) based on its real-time learning state and incorporate it into a unified loss function:

$$\mathcal{L}_{\text{SLS}}^{(i)}(\theta_t) = (1 - \alpha_i)\mathcal{L}_{\text{SFT}}^{(i)}(\theta_t) + \alpha_i \mathcal{L}_{\text{RL}}^{(i)}(\theta_t) \tag{4}$$

The fusion weight α_i is defined as:

$$\alpha_i = \min\left(1,\ \lambda\left(e^{M_i} - 1\right)\frac{D_i}{D_{\max}}\right) \tag{5}$$

where $\lambda > 0$ is a scaling factor, and $D_{\max}$ is the theoretical upper bound of the dispersion under the current reward specification, used to normalize D_i across different task and reward settings. The design of α_i is guided by three principles:

- **Mastery-first**: α_i increases monotonically with M_i, preventing the premature introduction of intensive RL exploration before the model establishes effective reasoning capabilities;
- **Anti-reward hacking**: α_i curbs surface-level trajectory dispersion induced by auxiliary rewards during low-mastery stages, preventing RL optimization from deviating from core task;
- **RL Dominance**: α_i rapidly elevates the RL weight for high-mastery, high-dispersion samples to fully leverage the contrastive learning signals inherent in diverse trajectories.

Equation (5) precisely operationalizes these principles. When both M_i and D_i are low, $\alpha_i \to 0$, ensuring SFT dominance to help the model establish

foundational reasoning capabilities. When D_i is high but M_i remains low, the term $e^{M_i} - 1$ leverages its exponential characteristics to significantly attenuate the growth of α_i, preventing over-reliance on RL driven by superficial trajectory dispersion induced by auxiliary rewards, thereby averting optimization divergence from core task objectives. Conversely, when M_i becomes high, the exponential term grows rapidly, elevating α_i for high-dispersion samples to fully leverage contrastive signals inherent in diverse exploration trajectories, while mitigating performance degradation caused by excessive supervision.

Additionally, we introduce a dynamic sample filtering mechanism to enhance training efficiency: samples that achieve the maximum total reward across all G rollouts (i.e., $R_{i,j} = R_{\max}, \forall j$) are removed from the current training batch. From the SLS perspective, such samples satisfy $M_i = 1$ (complete mastery) and $D_i = 0$ (no exploration diversity). Continuing to apply SFT constraints risks introducing redundant supervision and inducing overfitting, while their homogeneous trajectories fail to provide effective gradient signals for RL.

4 Experimental Setup

Experimental Data The training set comprises 32k samples: 30k filtered instances from OpenR1-Math-220k (Hugging Face, 2025) (prompts from NuminaMath1.5 (LI et al., 2024), responses from DeepSeek-R1 (DeepSeek-AI et al., 2025)) and 2k samples from GSM8K (Cobbe et al., 2021).

We evaluate all methods across four math benchmarks, including AIME 2024, AIME 2025, AMC (Jia LI and Polu, 2024), and MATH-500 (Hendrycks et al., 2021), as well as two Out-Of-Distribution (OOD) benchmarks, namely MMLU-Pro (Wang et al., 2024) and GPQA-Diamond (Rein et al., 2024), denoted as GPQA. Given the relatively small test sets of AIME and AMC, we report avg@32 to ensure stability, while reporting pass@1 for the other benchmarks. All evaluations employ a temperature of 0.6 with option shuffling applied to multiple-choice tasks to mitigate contamination.

Training Configuration. Experiments use the Qwen2.5-Math series models (Yang et al., 2024) with a context length of 8,192. The training batch size is set to 32. For each training sample, we generate $G = 4$ trajectories; this generation process is accelerated by vLLM (Kwon et al., 2023) using a sampling batch size of 64×4 and a temperature of 1.0. The learning rate is set to 1×10^{-5} for SFT; for other methods, we adopt their respective configurations, selecting from $\{1 \times 10^{-6}, 5 \times 10^{-6}\}$. Hybrid and RL methods train for one epoch, while SFT trains for three. For Equation (5), we set $\lambda = 2$, where $D_{\max}$ denotes the theoretical upper bound of reward standard deviation (Eq. 18). All experiments are conducted on four identical NVIDIA A800-80GB GPUs.

Baseline Methods. Both Qwen2.5-Math-1.5B and 7B are used as base models, we compare SLS against: (1) *Single methods*: **SFT**; **GRPO** (Shao et al., 2024), group-wise relative policy optimization; **DAPO** (Yu et al., 2025), filtering out all-correct and all-incorrect samples from the rollout buffer. (2) *Hybrid methods*: **SFT-then-RL**, a two-stage pipeline where RL (GRPO) training follows an initial SFT phase; **LUFFY** (Yan et al., 2025), utilizing online and offline policies to guide model learning; **ReLIFT** (Ma et al., 2025), interleaved online fine-tuning for hardest questions; **CHORD** (Zhang et al., 2025c), fusing SFT-RL with preset global weights.

5 Experimental Results

5.1 Main Experimental Results

Performance on Qwen2.5-Math Series. As shown in Table 2, SLS outperforms all baseline methods, establishing new state-of-the-art (SOTA) results. Specifically, it achieves average in-distribution accuracies of 40.6% (1.5B) and 51.8% (7B), while obtaining the highest scores on both OOD benchmarks.Notably, SLS surpasses DAPO by effectively learning from "all-incorrect" samples via adaptive SFT supervision, whereas DAPO simply discards them. To further substantiate the architectural generalizability of SLS, we provide supplementary evaluations on the general-purpose

Model	GPU Hours		Demonstrations		Training Steps	
	1.5B	7B	1.5B	7B	1.5B	7B
SFT	6×4	20×4	32k	32k	3000	3000
GRPO	15×4	36×4	-	-	1000	1000
DAPO	10×4	25×4	-	-	364	396
LUFFY	20×4	47×4	32k	32k	1000	1000
ReLIFT	16×4	39×4	15k	9k	1000	1000
CHORD	24×4	56×4	32k	32k	1000	1000
SLS	17×4	44×4	17k	10k	771	574

Table 1: Comparison of resource requirements and training steps of methods.

Model	In-Distribution (Core Task)										OOD			
	MATH-500		AIME 24		AIME 25		AMC 23		Avg.		MMLU-Pro		GPQA	
	1.5B	7B	1.5B	7B	1.5B	7B	1.5B	7B	1.5B	7B	1.5B	7B	1.5B	7B
Qwen2.5-Math														
Base	38.6	67.0	4.2	16.7	2.6	7.0	26.4	49.7	18.0	35.1	18.5	34.1	21.7	28.3
Instruct	71.6	82.0	9.7	9.3	7.6	7.9	44.7	49.3	33.4	37.1	30.0	45.7	30.3	32.2
Single Method														
GRPO	68.7	82.6	10.0	17.5	5.2	12.8	46.5	59.5	32.6	43.1	27.1	50.0	24.2	36.3
DAPO	67.6	81.2	9.4	16.8	5.2	10.5	42.2	59.3	31.1	42.0	24.3	51.4	25.7	34.3
SFT	65.6	80.8	10.4	24.1	10.9	**22.1**	39.7	61.1	31.7	47.0	27.1	47.1	12.6	31.8
Hybrid Method														
LUFFY	75.4	84.2	13.1	17.9	8.8	14.4	50.5	63.4	37.0	45.0	34.3	52.9	29.3	37.4
SFT-then-RL	73.5	83.1	10.7	20.5	9.0	16.8	44.6	63.1	34.5	45.9	30.3	44.4	23.2	36.5
ReLIFT	74.0	84.6	11.4	23.7	9.3	17.3	48.7	66.8	35.9	48.1	27.1	51.4	24.2	35.9
CHORD	76.2	87.2	13.1	23.2	10.5	19.3	51.3	65.8	37.8	48.9	40.0	51.4	25.8	40.4
SLS	**80.2**	**90.0**	**14.8**	**26.9**	**13.3**	21.3	**54.2**	**68.8**	**40.6**	**51.8**	**42.9**	**54.3**	**30.8**	**42.4**

Table 2: Overall performance on four math benchmarks and two OOD benchmarks based on Qwen2.5-Math-1.5B and 7B. **Bold** and underlined denote the best and second-best results, respectively.

Qwen2.5-7B and the next-generation **Qwen3-4B-Base** in Table 7 of the Appendix.

Table 1 presents the training efficiency. Compared to one-epoch SFT-RL fusion baselines, SLS requires fewer training steps and utilizes less demonstration data. Despite reduced resource consumption, SLS achieves the best or second-best performance acros all individual benchmarks.

Analysis of Training Dynamics and Exploration Capabilities Figure 4 illustrates the Shannon entropy (Eq. 19) dynamics of RL and SLS during the training of Qwen2.5-Math-1.5B. In pure RL, entropy exhibits a steady decline, indicating diminishing exploration capability and a gradual collapse of the output distribution. In contrast, the entropy of SLS rises rapidly in the initial phase to adapt to the expanded context length and specific training tasks, subsequently stabilizing at a level significantly higher than that of the RL baseline. This sustained high-entropy strategy enables SLS to continuously explore new strategies and effectively expand its knowledge boundaries.

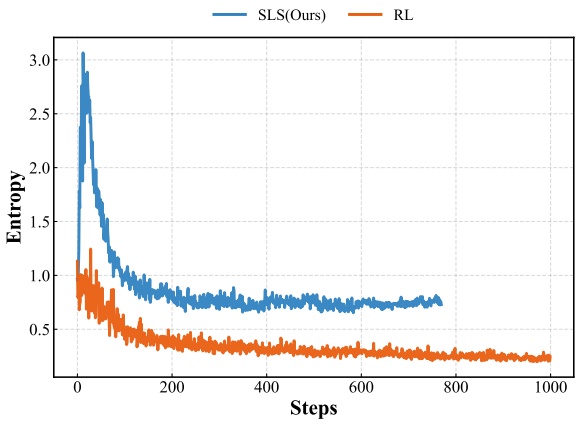

Figure 4: Training dynamics of entropy during RL and SLS training.

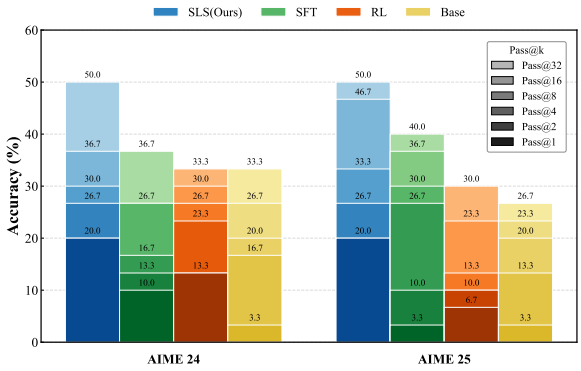

Figure 5: Pass@k performance on AIME 2024 and 2025. SLS achieves the best performance in both sampling efficiency (pass@1) and exploration capability and knowledge boundaries (pass@32).

Figure 5 compares the pass@k metrics across different methods and base model on the AIME 2024 and AIME 2025 benchmarks. The results indicate that SLS outperforms RL in pass@1 and exceeds SFT in pass@32, validating its superior sampling efficiency and broader knowledge boundary,

Setting	Model	In-Distribution (Core Task)					OOD		Aux.
		MATH	AIME24	AIME25	AMC	Avg. (Δ)	MMLU-Pro	GPQA	Score
Base	GRPO	70.0	10.0	5.2	46.5	32.9	27.1	24.2	–
	CHORD	76.2	13.1	10.5	51.3	37.8	40.0	25.8	–
	SLS	**80.2**	**14.8**	**13.3**	**54.2**	**40.6**	**42.9**	**30.8**	–
Format	GRPO	0.0	0.0	0.0	0.0	0.0 (-32.9)	0.0	0.0	90.9
	CHORD	58.2	5.5	7.2	38.4	27.3 (-10.5)	32.9	27.3	91.6
	SLS	**79.6**	**17.1**	**12.6**	**50.7**	**40.0** (-0.6)	**34.3**	**35.4**	**96.5**
Type	GRPO	67.4	8.8	5.6	44.0	31.5 (-1.4)	31.4	25.3	75.8
	CHORD	71.4	8.3	7.0	45.7	33.1 (-4.7)	28.6	25.8	32.3
	SLS	**80.4**	**14.7**	**13.3**	**54.2**	**40.7** (+0.1)	**42.9**	**28.8**	**79.8**

Table 3: Performance of multi-Reward. Parenthesized values indicate gaps relative to the *Base* setting. Zeros denote reward hacking.

respectively. These observations align with the theoretical analysis in Section 3.1: SLS achieves this through the synergistic regulation of sample mastery M and exploration dispersion D. Specifically, it strengthens SFT supervision for low-mastery samples to consolidate fundamental capabilities (high pass@1), while elevating RL weights for high-mastery, high-dispersion samples to extend knowledge boundaries (high pass@32), thereby achieving an optimal balance between sampling efficiency and exploration capability.

Robustness under Multi-Reward Settings To investigate the robustness of SLS in multi-objective optimization scenarios and verify the synergistic effectiveness of its core components (M and D), we conducted comparative experiments using Qwen2.5-Math-1.5B against GRPO (pure RL) and CHORD (static hybrid). We established three distinct reward configurations: (1) **Base**: utilizing solely the core task reward (r^{core}); (2) **Format**: augmenting the core reward with format compliance requirements ($r^{\text{core}} + r^{\text{format}}$); and (3) **Type**: incorporating a problem classification objective ($r^{\text{core}} + r^{\text{type}}$).

As detailed in Table 3, the introduction of secondary rewards poses significant challenges for baseline methods. The pure RL method, GRPO, is highly susceptible to reward hacking. In the Format setting, the model suffers a catastrophic collapse, where generated outputs degenerate into repetitive prompt segments, causing core reasoning accuracy to plummet to zero across all tasks. Even in the Type setting, where no explicit hacking is observed, interference from auxiliary signals degrades the average core accuracy from 32.9% to 30.5%.

Although the static SFT-RL hybrid method, CHORD, partially mitigates this instability through supervised regularization, it fails to effectively balance competing objectives. Its core performance regresses suffers significantly regression (dropping from 37.8% to 27.3% in the Format setting). Furthermore, CHORD struggles to optimize the auxiliary objective in the Type setting, defaulting to a trivial solution by assigning all questions to a single category.

In contrast, SLS demonstrates remarkable robustness, maintaining stable core performance across all three scenarios (Base: 40.6%, Format: 40.0%, Type: 40.3%). Simultaneously, it achieves superior alignment with auxiliary preferences, attaining scores of 96.5% and 79.8% on Format and Type tasks, respectively, with no evidence of reward hacking.

These empirical results validate the theoretical design of our dynamic fusion mechanism (Section 3). Specifically, in scenarios characterized by high dispersion (D) but low mastery (M)—typically involving superficial variations induced by secondary rewards—the exponential damping factor ($e^{M_i} - 1$) effectively suppresses the RL weight. This prevents the optimizer from exploiting easy-to-hack auxiliary signals at the expense of reasoning logic, ensuring a precise balance between core task mastery and preference alignment.

5.2 Ablation Study

Impact of Different α Configurations. We perform ablation studies on Qwen2.5-Math-1.5B under the susceptible $r^{\text{core}} + r^{\text{format}}$ reward setting to isolate the contributions of each component:

As shown in Table 4, the full SLS model achieves superior performance (40.0% Avg.) compared to all variants. The Static and Random

Model	In-Distribution					OOD		Aux.
	MATH	AIME24	AIME25	AMC	Avg.	MMLU-Pro	GPQA	Format
Random	73.5	9.1	9.4	47.5	34.9	18.9	23.2	90.7
Static	75.8	11.2	9.3	45.0	35.3	27.5	21.2	87.3
M-only	78.4	14.9	12.4	**52.1**	39.5	30.7	23.4	94.5
D-only	77.5	13.2	12.1	48.0	37.7	31.2	24.1	95.4
SLS	**79.6**	**17.1**	**12.6**	50.7	**40.0**	**42.9**	**28.8**	**96.5**

Table 4: Performance of different α configurations. Static: Fixed $\alpha_i = 0.5$; Random: $\alpha_i \sim \mathcal{U}(0, 1)$; M-only: $\alpha_i = \lambda(e^{M_i} - 1)$; D-only: $\alpha_i = \lambda \frac{D_i}{D_{\max}}$

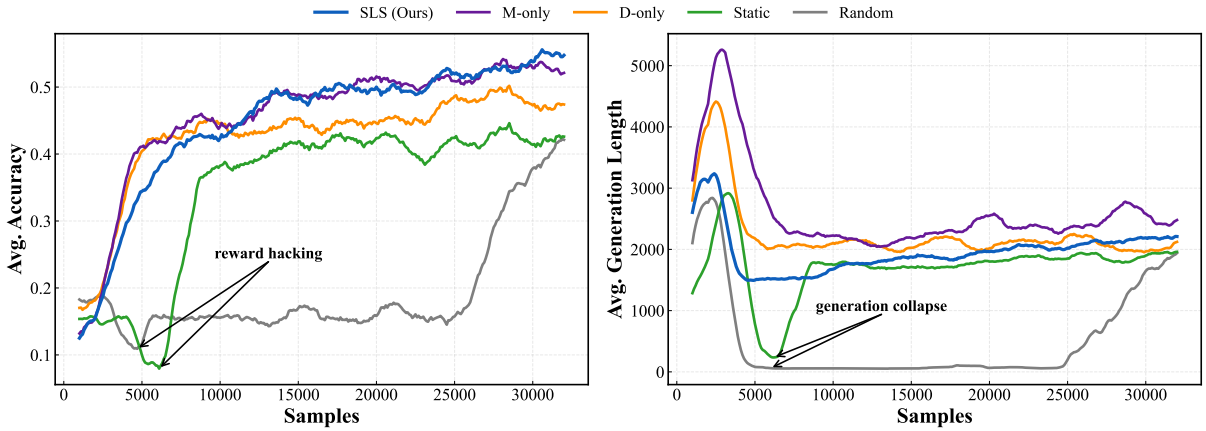

Figure 6: Training dynamics of different α configurations. **Left:** Average Accuracy on the core task. **Right:** Average Generation Length. Static and Random baselines suffer from severe length collapse (indicative of reward hacking),whereas SLS ensures stable performance improvement.

baselines succumb to catastrophic reward hacking, evidenced by sharp output length collapse and core task degradation (Figure 6), confirming that weights independent of sample learning states cannot adapt to sample heterogeneity and are easily misled by superficial trajectory differences.

Crucially, the results validate the necessity of the dual-dimensional SLS design: the M-only variant (39.5%) neglects the exploration dispersion, forfeiting RL gains from high-trajectory-variance samples, which compromises OOD generalization and secondary objective performance; while, the D-only variant (37.7%) lacks the mastery constraint, prematurely amplifying RL weights on low-mastery samples. This induces a deviation in optimization objectives (achieving a secondary score of 95.4% despite the regression of core performance), demonstrating that both dimensions are indispensable for robust dynamic fusion.

Impact of Scaling Factor λ. Table 5 in the Appendix demonstrates that SLS maintains robust performance across settings, with $\lambda = 2$ striking the optimal balance between supervision and exploration. Specifically, lower values ($\lambda = 1$) result in an SFT-dominated regime, constraining both OOD generalization and the optimization of secondary preferences. Conversely, higher values ($\lambda = 3$) amplify the RL influence; while SLS effectively prevents severe reward hacking even in this setting, the excessive exploration compromises stability and degrades core task performance.

6 Conclusions

The paper proposes Sample Learning State (SLS), which characterizes each sample's learning state using the degree of sample mastery (M_i) and the dispersion of exploration trajectory (D_i). We derive a model-state-aware, sample-wise dynamic weighting coefficient (α_i) to adaptively integration of SFT and RL, balancing supervised guidance and autonomous exploration. Experiments show that our method achieves new SOTAs with fewer training steps and mitigates reward hacking. Future work will extend SLS to broader domains and more complex instruction-alignment scenarios to assess its cross-domain generality and scalability.

7 Limitations

Our method is primarily developed for RL with verifiable rewards (RLVR), where the sample mastery score M relies on deterministic binary core rewards. In more common RLHF or preference-alignment settings, rewards are typically continuous and noisy, so the stability and interpretability of SLS remain to be validated. Moreover, since RLVR provides only outcome-level signals, SLS cannot assess the quality of intermediate reasoning processes along exploration trajectories; when a model reaches a correct answer via unreliable reasoning, the sample may be mistakenly treated as "mastered," which can affect learning of core reasoning skills. In addition, our training data and main gains are concentrated in reasoning tasks, and extending SLS to more complex instruction-alignment scenarios without objective verification is left for future work. Finally, our experiments mainly cover 1.5B and 7B models; the behavior of SLS at larger scales and its interaction with scaling trends require further study.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Hardy Chen, Haoqin Tu, Fali Wang, Hui Liu, Xianfeng Tang, Xinya Du, Yuyin Zhou, and Cihang Xie. 2025a. Sft or rl ? an early investigation into training r1-like reasoning large vision-language models. *arXiv preprint arXiv:2504.11468*.

Jack Chen, Fazhong Liu, Naruto Liu, Yuhan Luo, Erqu Qin, Harry Zheng, Tian Dong, Haojin Zhu, Yan Meng, and Xiao Wang. 2025b. Step-wise adaptive integration of supervised fine-tuning and reinforcement learning for task-specific llms. *Preprint*, arXiv:2505.13026.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. 2025. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. In *Forty-second International Conference on Machine Learning*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, and 1 others. 2025. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Hugging Face. 2024. Math-verify. `https://github.com/huggingface/Math-Verify`. Accessed: 2024-06-10.

Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. 2025. Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning. *Preprint*, arXiv:2506.19767.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Lewis Tunstall Ben Lipkin Roman Soletskyi Shengyi Costa Huang Kashif Rasul Longhui Yu Albert Jiang Ziju Shen Zihan Qin Bin Dong Li Zhou Yann Fleureau Guillaume Lample Jia LI, Edward Beeching and Stanislas Polu. 2024. Numinamath. `[https://github.com/project-numina/aimo-progress-prize](https://github.com/`

project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf).

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

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-1.5](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf).

Zhihan Liu, Miao Lu, Shenao Zhang, Boyi Liu, Hongyi Guo, Yingxiang Yang, Jose Blanchet, and Zhaoran Wang. 2024. Provably mitigating overoptimization in RLHF: Your SFT loss is implicitly an adversarial regularizer. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Xingtai Lv, Yuxin Zuo, Youbang Sun, Hongyi Liu, Yuntian Wei, Zhekai Chen, Lixuan He, Xuekai Zhu, Kaiyan Zhang, Bingning Wang, and 1 others. 2025. Towards a unified view of large language model post-training. *arXiv preprint arXiv:2509.04419*.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, Bin Cui, and Wentao Zhang. 2025. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *Preprint*, arXiv:2506.07527.

Phuc Minh Nguyen, Chinh D La, Duy MH Nguyen, Nitesh V Chawla, Binh T Nguyen, and Khoa D Doan. 2025. The reasoning boundary paradox: How reinforcement learning constrains language models. *arXiv preprint arXiv:2510.02230*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and characterizing reward hacking. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021.

Mingyu Su, Jian Guan, Yuxian Gu, Minlie Huang, and Hongning Wang. 2025. Trust-region adaptive policy optimization. *arXiv preprint arXiv:2512.17636*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yongliang Wu, Yizhou Zhou, Zhou Ziheng, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. 2025. On the generalization of sft: A reinforcement learning perspective with reward rectification. *arXiv preprint arXiv:2508.05629*.

Hongling Xu, Qi Zhu, Heyuan Deng, Jinpeng Li, Lu Hou, Yasheng Wang, Lifeng Shang, Ruifeng Xu, and Fei Mi. 2025. Kdrl: Post-training reasoning llms via unified knowledge distillation and reinforcement learning. *Preprint*, arXiv:2506.02208.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. 2025. Learning to reason under off-policy guidance. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2025. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, Yu Fu, Xingtai Lv, Yuchen Zhang, Sihang Zeng, Shang Qu, Haozhan Li, Shijie Wang, Yuru Wang, Xinwei Long, and 20 others. 2025a. A survey of reinforcement learning for large reasoning models. *Preprint*, arXiv:2509.08827.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Guoyin Wang, and Fei Wu. 2025b. Instruction tuning for large language models: A survey. *ACM Comput. Surv.* Just Accepted.

Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin Ding, and Jingren Zhou. 2025c. On-policy rl meets off-policy experts: Harmonizing supervised fine-tuning and reinforcement learning via dynamic weighting. *CoRR*, abs/2508.11408.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: Less is more for alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*.

A Experimental Details

Dataset. We construct the training set by combining 30k filtered samples from **OpenR1-Math-220k** (Hugging Face, 2025) and 2k randomly sampled instances from **GSM8K** (Cobbe et al., 2021), mixed uniformly unless otherwise stated. For OpenR1-Math, we filter samples by (i) answer correctness verified with `math-verify` (Face, 2024), (ii) format compliance, (iii) length (< 8192 tokens), and (iv) the absence of hyperlinks.

Models. For the Qwen2.5-Math models, we follow (Yan et al., 2025) and apply RoPE scaling by setting the RoPE θ from 10,000 to 40,000, which enables a larger context window (16,384). Unless otherwise stated, we cap the maximum generation length at 8,192 tokens. We additionally evaluate on **Qwen2.5-7B** and **Qwen3-4B-Base** (Yang et al., 2025).

Hyperparameters. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and no weight decay. We adopt a constant learning rate schedule with zero warmup steps. The learning rate is 1×10^{-5} for SFT, 1×10^{-6} for GRPO/DAPO/LUFFY/ReLIFT, and 5×10^{-6} for CHORD and SLS. For RL-based methods, we sample trajectories with temperature 1.0. We set the per-update prompt batch size to 32, and generate $G = 4$ rollouts per prompt, resulting in an effective rollout batch of 32×4 **trajectories** per update.

Evaluation. We use a unified evaluation protocol: temperature 0.6 and max generation length 8,192. Answer correctness is verified by `math-verify`. All baselines are evaluated under the same settings.

B System Prompts

We utilize task-specific system prompts. The same prompts are used for both training and evaluation to ensure consistency.

Core Task Only.

```
You are a helpful AI assistant that provides detailed thought processes and correct answers. Your final answer must be placed within \boxed{}.
```

With Format Preference.

```
You are a helpful AI Assistant that provides detailed thought processes and correct answers. Please respond in the following format:
<think>
[Your step-by-step thinking process here]
</think>
<answer>
[Your clear solution and answer here, with the answer in \boxed{}]
</answer>
```

With Problem Type Preference.

```
You are a helpful AI assistant that provides detailed thought processes and correct answers. Your final answer must be placed within \boxed{}.
After answering, select the most appropriate category from: Algebra, Geometry, Number Theory, Combinatorics, Primary Math, Calculus, Logic and Puzzles, Inequalities, or Other. End your response strictly with: This is a [Category] question.
```

C Reward Functions

Unless otherwise specified, the following reward configurations are applied consistently across all training experiments.

Core Task Reward. We employ a standard binary outcome reward for mathematical reasoning:

$$r^{\text{core}} = \begin{cases} 1, & \text{if correct} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Format Compliance Reward. To enforce a "Think-then-Answer" reasoning paradigm, we mandate that outputs strictly adhere to the structural format: `<think>...</think><answer>...</answer>`. Furthermore, to alleviate the issue of reward sparsity inherent in binary scoring—particularly for smaller models—we formulate a fine-grained dense reward function:

$$r^{\text{format}} = \sum_{n=1}^{4} w_n \cdot \mathbb{I}(C_n) \tag{7}$$

where $\mathbb{I}(\cdot)$ is the indicator function, and the components are defined as:

$$\begin{aligned} w_1 &= 0.075, \quad C_1 : \text{Tag existence (per tag)} \\ w_2 &= 0.10, \quad C_2 : \text{Tag uniqueness (per tag)} \\ w_3 &= 0.10, \quad C_3 : \text{Order of adjacent pairs} \\ w_4 &= 1.00, \quad C_4 : \text{Full answer wrapping} \end{aligned}$$

Format tags are inherent in the OpenR1 dataset and were manually annotated for GSM8K.

Problem Type Reward. To simulate auxiliary preference alignment, we introduce a classification task using OpenR1 problem types as ground truth labels (defaulting to "Primary Math" for GSM8K). The reward is defined as:

$$r^{\text{type}} = \begin{cases} 1, & \text{if prediction matches label} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

D Settings for Motivation Experiments

This section details the experimental configurations for the motivational analyses presented in Section 1 (Figure 1) and Section 3 (Figure 2 and Figure 3).

D.1 Dataset Construction and Difficulty Grouping

We utilize the full training sets of OpenR1-Math and GSM8K as the data source. Using the Qwen2.5-Math-7B model, we perform $G = 4$ rollouts for each sample. Based on the correctness ratio, we stratify the samples into five mastery levels: Level-0 for $[0, 0.2)$, Level-1 for $[0.2, 0.4)$, Level-2 for $[0.4, 0.6)$, Level-3 for $[0.6, 0.8)$, and Level-4 for $[0.8, 1.0]$.

To ensure unbiased evaluation, we randomly sample 100 instances from each level to construct a balanced test set consisting of 500 samples in total. This set is strictly isolated from all training processes. All evaluations regarding secondary preferences are conducted on this dataset. The remaining data constitute the training set for the specific experiments described below.

D.2 Setup for Static Weight Analysis

Figure 1 investigates static SFT and RL weights by evaluating core task capability on the held-out test set. We train Qwen2.5-Math-7B on the shuffled dataset incorporating format rewards, where an online replay buffer accumulates filtered trajectories for updates. The loss function follows Equation 4, compared across two configurations:

- **Config 1 ($\alpha = 0.1$):** SFT-dominant (High SFT weight, Low RL weight).
- **Config 2 ($\alpha = 0.9$):** RL-dominant (Low SFT weight, High RL weight).

We run training for 120 update steps and save a checkpoint every 20 steps for evaluation. Following Section 1, we define easy samples as having at least 3 correct rollouts out of 4, and hard samples as having at most 1 correct rollout.

D.3 Setup for Analysis of the Degree of Sample Mastery $M_i(t)$

Figure 2 illustrates the relationship between the degree of sample mastery ($M_i(t)$) and SFT effectiveness by quantifying performance gains on the test set. We conduct five independent SFT experiments, each utilizing one of the disjoint data subsets corresponding to a specific mastery level (Level-0 to Level-4), while maintaining hyperparameters consistent with the main SFT baseline.

D.4 Setup for Analysis of the Dispersion of Exploration Trajectories $D_i(t)$

Figure 3 investigates the impact of the dispersion of exploration trajectories ($D_i(t)$) on RL optimization effectiveness. Specifically, we monitor the total reward and core task accuracy on the held-out test set. We conduct online training using a shuffled dataset with both format and core task rewards enabled, evaluating performance across three distinct dispersion intervals (*High*, *Medium*, and *Low*).

E Formula Details

In this section, we detail the mathematical formulations for the SFT loss, the GRPO objective, and the specific metrics used in our Sample Learning State framework.

Supervised Fine-Tuning (SFT) Loss. For a given sample $s_i = (x_i, y_i)$, the SFT loss minimizes the negative log-likelihood of the reference reasoning sequence y_i. Let $y_{i,<l}$ denote the tokens preceding position l:

$$\mathcal{L}_{\text{SFT}}^{(i)}(\theta_t) = -\frac{1}{|y_i|}\sum_{l=1}^{|y_i|}\log \pi_{\theta_t}(y_{i,l} \mid x_i, y_{i,<l}) \quad (9)$$

Dispersion of Exploration Trajectories ($D_i(t)$). Following Section 3.1, we measure trajectory dispersion for sample s_i at step t by the standard deviation of the composite rewards across G independent rollouts:

$$D_i(t) = \text{Std}\big(\{R_{i,j}\}_{j=1}^{G}\big), \quad (10)$$

where the composite reward for the j-th trajectory is

$$R_{i,j} = r_{i,j}^{\text{core}} + \sum_{k=1}^{K} r_{i,j}^{k}. \quad (11)$$

the standard deviation is computed as

$$\begin{aligned}\text{Std}\big(\{R_{i,j}\}_{j=1}^{G}\big) &= \sqrt{\frac{1}{G}\sum_{j=1}^{G}\big(R_{i,j} - \bar{R}_i\big)^2}, \\ \bar{R}_i &= \frac{1}{G}\sum_{j=1}^{G} R_{i,j}.\end{aligned} \quad (12)$$

Reinforcement Learning (RL) Loss. We employ the Group Relative Policy Optimization (GRPO) objective (Shao et al., 2024) as our RL loss function. Following (Cui et al., 2025; Yu et al., 2025; Hu et al., 2025), we remove the KL-divergence regularization term to avoid imposing additional constraints on policy update magnitudes, thereby enabling more aggressive exploration in the early training stages.

For sample s_i with input x_i, we generate a group of G output trajectories $\mathcal{O}_i = \{o_{i,1}, \ldots, o_{i,G}\}$ from the current policy $\pi_{\theta_{old}}$. The GRPO objective with PPO-style clipping is defined as:

$$\begin{aligned}\mathcal{L}_{\text{RL}}^{(i)}(\theta_t) = -\frac{1}{G}\sum_{j=1}^{G}\min\big(&\rho_{i,j}\hat{A}_{i,j}, \\ &\text{clip}(\rho_{i,j}, 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}})\hat{A}_{i,j}\big)\end{aligned} \quad (13)$$

where $\epsilon_{\text{clip}} = 0.2$ is the clipping parameter, and the importance sampling ratio $\rho_{i,j}$ is computed as:

$$\rho_{i,j} = \frac{\pi_{\theta_t}(o_{i,j} \mid x_i)}{\pi_{\theta_{old}}(o_{i,j} \mid x_i)} \quad (14)$$

Model	In-Distribution (Core Task)					OOD		Aux.	Training
	MATH	AIME24	AIME25	AMC23	Avg.	MMLU-Pro	GPQA	Format	Step
$\lambda = 1$	<u>78.6</u>	<u>14.5</u>	<u>12.9</u>	<u>50.7</u>	<u>39.2</u>	<u>38.5</u>	<u>26.3</u>	94.4	<u>855</u>
$\lambda = 3$	69.8	11.1	9.9	40.6	32.9	32.9	14.1	<u>95.7</u>	913
SLS ($\lambda = 2$)	**80.2**	**14.8**	**13.3**	**54.2**	**40.6**	**42.9**	**30.8**	**96.5**	**779**

Table 5: Impact of λ on Qwen2.5-Math-1.5B under the $r^{\text{core}} + r^{\text{format}}$ setting. **Bold** and <u>underlined</u> denote the best and second-best results, respectively.

Mixing Strategy	In-Distribution (Core Task)					OOD		Training
	MATH	AIME24	AIME25	AMC23	Avg.	MMLU-Pro	GPQA	Steps
Random Sampling	79.6	16.4	12.2	54.0	40.6	40.0	30.3	744
Curriculum Learn	79.8	14.7	13.1	54.6	40.6	38.5	27.8	754
Uniform (SLS)	80.2	14.8	13.3	54.2	40.6	42.9	30.8	771

Table 6: Impact of data mixing configurations on Qwen2.5-Math-1.5B. SLS exhibits high robustness across different sampling strategies, consistently achieving optimal performance on OOD benchmarks.

and $\hat{A}_{i,j}$ is the advantage, computed using the group of rewards $\{R_{i,1}, R_{i,2}, \ldots, R_{i,G}\}$ corresponding to the trajectories within each group:

$$\hat{A}_{i,j} = \frac{R_{i,j} - \text{mean}(\{R_{i,1}, R_{i,2}, \cdots, R_{i,G}\})}{\text{std}(\{R_{i,1}, R_{i,2}, \cdots, R_{i,G}\})} \tag{15}$$

Degree of Sample Mastery ($M_i(t)$). As defined in Section 3.1, we compute the mastery of sample s_i at step t as the average core reward over G rollouts:

$$M_i(t) = \frac{1}{G} \sum_{j=1}^{G} \tilde{r}_{i,j}^{\text{core}}, \tag{16}$$

where $\tilde{r}_{i,j}^{\text{core}} \in [0, 1]$ denotes the normalized core reward. If the original core reward already lies in $[0, 1]$ (e.g., $r_{i,j}^{\text{core}} \in \{0, 1\}$ in RLVR), we set $\tilde{r}_{i,j}^{\text{core}} = r_{i,j}^{\text{core}}$. Otherwise, we map $r_{i,j}^{\text{core}}$ to $[0, 1]$ via threshold-based normalization with $\tau_{\text{low}} < \tau_{\text{high}}$, where rewards below τ_{low} are treated as "not mastered" and rewards above τ_{high} as "mastered":

$$\tilde{r}_{i,j}^{\text{core}} = \begin{cases} 0, & r_{i,j}^{\text{core}} < \tau_{\text{low}}, \\ \dfrac{r_{i,j}^{\text{core}} - \tau_{\text{low}}}{\tau_{\text{high}} - \tau_{\text{low}}}, & \tau_{\text{low}} \le r_{i,j}^{\text{core}} < \tau_{\text{high}}, \\ 1, & r_{i,j}^{\text{core}} \ge \tau_{\text{high}}. \end{cases} \tag{17}$$

This makes $M_i(t)$ comparable across different core-reward specifications.

$D_{\max}$ for normalization. In Equation (5), we normalize D_i by $D_{\max}$. Let $R_{i,j} \in [R_{\min}, R_{\max}]$ and define $\Delta = R_{\max} - R_{\min}$. We set

$$D_{\max} = \Delta \sqrt{\frac{\lceil G/2 \rceil \lfloor G/2 \rfloor}{G^2}}. \tag{18}$$

Shannon Entropy (H). As an auxiliary indicator of output uncertainty, we compute the mean token-level Shannon entropy for the j-th rollout trajectory $o_{i,j} = (o_{i,j,1}, \ldots, o_{i,j,|o_{i,j}|})$ of sample s_i under the current policy π_{θ_t}:

$$H(o_{i,j}) = -\frac{1}{|o_{i,j}|} \sum_{l=1}^{|o_{i,j}|} \sum_{v \in \mathcal{V}} \pi_{\theta_t}(v \mid x_i, o_{i,j,<l}) \cdot \log \pi_{\theta_t}(v \mid x_i, o_{i,j,<l}) \,. \tag{19}$$

where $\mathcal{V}$ is the vocabulary and $o_{i,j,<l}$ denotes the prefix tokens of $o_{i,j}$.

F Additional Experimental Results

Impact of the scaling factor λ. We evaluate the effect of the scaling factor λ in Eq. 5 under the multi-reward setting $r^{\text{core}} + r^{\text{format}}$. Table 5 reports the results. Since λ controls the overall magnitude of the fusion weight α_i, smaller values place more emphasis on SFT, while larger values increase the contribution of RL. We observe that $\lambda = 2$ achieves the best overall trade-off, with the highest in-distribution average accuracy and the strongest OOD performance, while also requiring fewer training steps. In contrast, $\lambda = 1$ performs slightly worse on the core and OOD benchmarks, and $\lambda = 3$ increases the auxiliary format score but degrades core and OOD performance.

Impact of data mixing configurations. The two training sources differ substantially in difficulty: on the base model, OpenR1-Math-Default attains 12.28% accuracy, while GSM8K reaches 57.43%.

Model	In-Distribution (Core Task)					OOD	
	MATH	AIME24	AIME25	AMC23	Avg.	MMLU-Pro	GPQA
Qwen2.5-7B							
Qwen2.5-7B	57.4	7.1	4.8	36.7	26.5	44.2	25.8
Qwen2.5-7B-Instruct	<u>77.0</u>	10.5	7.1	50.8	36.4	<u>62.9</u>	<u>34.3</u>
SFT	69.8	<u>12.6</u>	**14.1**	40.0	34.1	50.0	18.1
RL	75.4	12.1	6.6	<u>52.0</u>	<u>36.5</u>	57.6	33.8
SLS (Ours)	**82.0**	**15.0**	<u>12.8</u>	**56.9**	**41.7**	**64.3**	**34.8**
Qwen3-4B-Base							
Qwen3-4B-Base	69.4	9.7	7.6	45.0	32.9	54.2	32.8
SFT	75.0	16.8	**19.1**	51.4	40.6	62.9	24.7
RL	<u>82.6</u>	<u>19.0</u>	15.1	<u>60.1</u>	<u>44.2</u>	<u>68.6</u>	<u>39.9</u>
SLS (Ours)	**84.4**	**19.1**	<u>16.3</u>	**62.3**	**45.5**	**72.9**	**42.4**

Table 7: Results on additional base models. **Bold** and <u>underlined</u> denote the best and second-best results, respectively.

To test whether SLS is sensitive to how these sources are combined, we compare three mixing strategies: (1) random sampling, (2) curriculum learning (from easier to harder samples), and (3) uniform mixing. Table 6 shows that SLS yields comparable in-distribution performance across all strategies (Avg. 40.6%), with only minor variations on OOD benchmarks and training steps. Unless otherwise stated, we use uniform mixing in all experiments. These results suggest that the sample-wise dynamic weighting in SLS can adapt to heterogeneous data without changing the mixing schedule.

Results on additional base models. To assess the generality of SLS beyond the Qwen2.5-Math family, we further evaluate it on a general-purpose model (Qwen2.5-7B) and a newer base model (Qwen3-4B-Base). As shown in Table 7, SLS achieves the best OOD performance on Qwen2.5-7B, while remaining competitive on in-distribution benchmarks. On Qwen3-4B-Base, SLS attains the best in-distribution average accuracy and the best OOD scores, outperforming both SFT and RL. Overall, these results suggest that SLS transfers across model families.