Not All Preferences Are Created Equal: Stability-Aware and Gradient-Efficient Alignment for Reasoning Models

Anonymous ACL submission

Abstract

Preference-based alignment is pivotal for training large reasoning models; however, standard methods like Direct Preference Optimization (DPO) typically treat all preference pairs uniformly, overlooking the evolving utility of training instances. This static approach often leads to inefficient or unstable optimization, as it wastes computation on trivial pairs with negligible gradients and suffers from noise induced by samples near uncertain decision boundaries. Facing these challenges, we propose **SAGE** (Stability-Aware Gradient Efficiency), a dynamic framework designed to enhance alignment reliability by maximizing the Signal-to-Noise Ratio of policy updates. Concretely, SAGE integrates a coarse-grained curriculum mechanism that refreshes candidate pools based on model competence with a fine-grained, stability-aware scoring function that prioritizes informative, confident errors while filtering out unstable samples. Experiments on multiple mathematical reasoning benchmarks demonstrate that SAGE significantly accelerates convergence and outperforms static baselines, highlighting the critical role of policy-aware, stability-conscious data selection in reasoning alignment.

1 Introduction

With the emergence of large reasoning models such as DeepSeek-R1 (Guo et al., 2025) and OpenAI-o1 (Jaech et al., 2024), significant progress has been made in addressing complex reasoning tasks. These models not only output final answers but also generate explicit reasoning trajectories, referred to as long chains-of-thought (Long-CoT). These trajectories often yield verifiable outcomes, which can be used as supervision signals for reinforcement learning (Shao et al., 2024) or transformed into pairwise preferences for alignment methods such as Direct Preference Optimization (DPO) (Rafailov et al., 2023). As DPO effectively models preferences through response likelihoods without requiring explicit reward modeling, it is widely adopted in the early stages of aligning reasoning models.

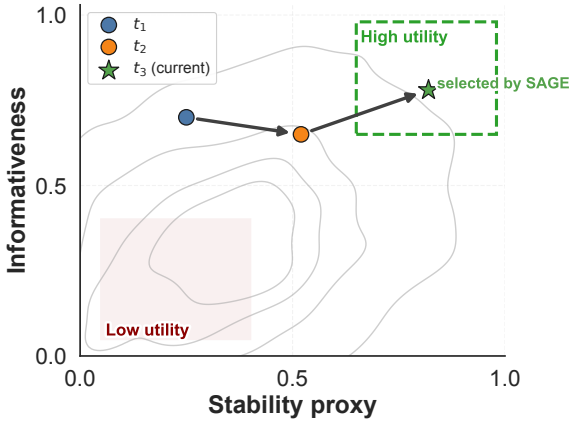


Figure 1: Schematic illustration of dynamic sample utility in the stability-informativeness space. SAGE prioritizes samples with high signal quality and stable optimization behavior.

However, standard preference optimization often treats all data pairs uniformly, overlooking critical disparities in their learning value. First, from an efficiency perspective, strictly easier or "thin-margin" pairs yield negligible gradients, contributing little to the optimization while consuming computational budget. Second, and more critically, from a stability perspective, pairs near the model's unstable decision boundaries often exhibit high local curvature in the loss landscape. In these regions, small perturbations can induce disproportionately large, high-variance gradients. Naively training on such "noisy" data can destabilize the alignment of reasoning models, where maintaining precise logical chains is essential.

Furthermore, the utility of a preference pair is not an inherent property but is dynamically coupled to the model's evolving state; a sample providing strong supervision early in training may offer diminishing returns later as the model internalizes

the underlying reasoning pattern. Consequently, static data selection is suboptimal.

To overcome these challenges, we formulate preference selection from an optimization perspective and propose **S**tability-**A**ware **G**radient **E**fficiency (**SAGE**), a policy-aware alignment framework that adapts supervision to the evolving policy state by jointly considering gradient signal and prediction confidence. **First**, SAGE introduces a coarse-grained, pool-based on-policy approximation that partitions the static corpus into refreshable and disjoint pools, and re-scores only the active pool with the current policy. This design preserves policy adaptivity while avoiding full-dataset rescoring, and its scheduling mechanism naturally forms a curriculum over preference difficulty. **Second**, SAGE performs fine-grained sample selection within each pool using a stability-aware utility score that balances gradient magnitude with prediction confidence. This score prioritizes reliable, high-signal confident errors while filtering out weak-signal or unstable pairs, resulting in more stable and gradient-efficient policy updates. Together, these components provide a dynamic and computationally efficient preference-selection framework that focuses training on the most useful supervision signals at each stage of alignment.

Our contributions are summarized as follows:

- We identify that the learning value of reasoning preference pairs is dynamic and constrained by local loss geometry, revealing that static training overlooks both gradient efficiency and optimization stability.
- We propose SAGE, a policy-adaptive alignment framework that achieves dynamic preference selection by periodically re-scoring refreshable pools under the current model state.
- We introduce a stability-aware utility score that enables fine-grained filtering by balancing the gradient signal with a proxy for local prediction confidence, allowing the model to prioritize reliable and informative supervision.
- Experiments on Long-CoT math reasoning benchmarks demonstrate that SAGE improves alignment stability, sample efficiency, and overall performance.

2 Preliminaries

Notation. Let x denote a prompt and y a response. We define a trainable policy by $\pi_\theta(y \mid x)$ and a fixed reference policy $\mu(y \mid x)$. Preference supervision consists of triples (x, y^w, y^l), where $y^w \succ y^l$ indicates that y^w is preferred over y^l for the same prompt x. In the context of mathematical reasoning, x represents a problem statement and y a candidate solution. Typically, y^w corresponds to a correct or higher-quality answer, whereas y^l is incorrect or of lower quality.

Direct Preference Optimization (DPO). DPO optimizes π_θ directly from preference data without relying on an explicit reward model or on-policy reinforcement learning. Formally, DPO defines an implicit reward:

$$r_\theta(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\mu(y \mid x)} \tag{1}$$

where $\beta > 0$ is a temperature parameter. The training objective follows a Bradley–Terry likelihood:

$$\mathcal{L}_{\mathrm{DPO}} = -\,\mathbb{E}_{(x,y^w,y^l)} \log \sigma\big(r_\theta(x, y^w) - r_\theta(x, y^l)\big) \tag{2}$$

with $\sigma(\cdot)$ denoting the sigmoid function.

Noise-Contrastive Alignment (NCA). NCA extends preference optimization to a multi-response contrastive setting. Given a prompt x with K candidate responses $\{y_i\}_{i=1}^K$, it computes normalized weights:

$$w_i(x) = \frac{\exp(r_i/\alpha)}{\sum_{j=1}^K \exp(r_j/\alpha)} \tag{3}$$

where $r_i = r_\theta(x, y_i)$ and $\alpha > 0$ is a temperature. The NCA objective is defined as:

$$\mathcal{L}_{\mathrm{NCA}} = -\sum_{i=1}^{K} \left[w_i(x)\, \log \sigma(r_i) + \frac{1}{K}\, \log \sigma(-r_i) \right] \tag{4}$$

Compared to DPO, NCA encourages high-reward responses while regularizing uniformly weighted negatives, improving training stability. Structurally, the NCA loss is a weighted sum of binary classification components (one positive, K negatives), which allows us to analyze the optimization dynamics of each response independently. When $K{=}2$ and the weight $w_i(x)$ concentrates on the preferred response, this formulation reduces to the DPO objective structure. Unless otherwise specified, the loss $\mathcal{L}$ in subsequent sections refers to the generic pairwise preference objective.

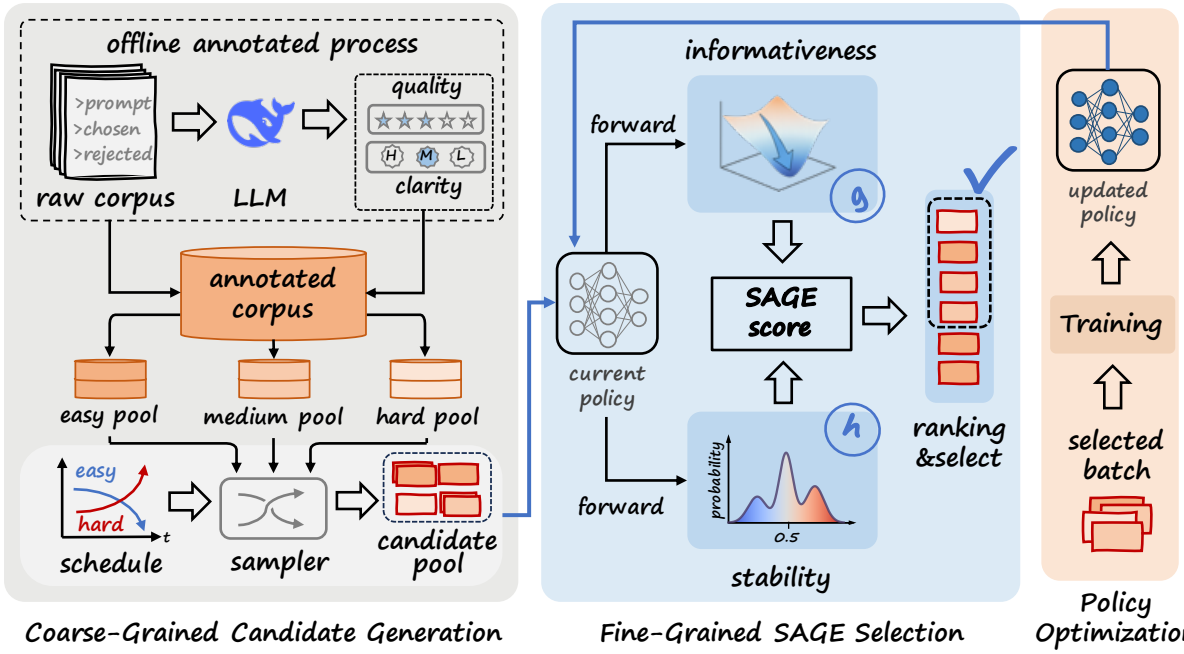


Figure 2: Overall framework of SAGE.

3 Method

This section introduces **S**tability-**A**ware **G**radient **E**fficiency (**SAGE**), a dynamic alignment framework designed to optimize the learning progress of reasoning models. Unlike static strategies that treat all preference pairs equally, SAGE formulates data selection as a policy-adaptive prioritization problem, motivated by improving the signal quality of supervision. It dynamically prioritizes samples that provide strong gradient signals while avoiding unstable updates under the evolving policy. This approach unfolds in two phases, as illustrated in Figure 2: first, generating coarse-grained candidate pools based on difficulty priors (Section 3.2), followed by fine-grained selection to maximize gradient efficiency and ensure optimization stability (Section 3.3).

3.1 Problem Setup

We consider the alignment of a reasoning policy π_θ using a static dataset $\mathcal{D} = \{(x_i, y_i^w, y_i^l)\}_{i=1}^N$, where x_i is a prompt and (y_i^w, y_i^l) are the winning and losing responses (see Appendix A.1 for data construction). The standard objective minimizes a loss $\mathcal{L}(\theta)$ over the expectation of the data. However, minimizing the average loss over a static distribution ignores the dynamic nature of sample utility. Our goal is to dynamically select a sub-batch $\mathcal{B}_t \subset \mathcal{D}$ at each step t that maximizes the optimization gain.

Remark on On-Policy Selection. It is crucial to clarify our terminology. In this work, *on-policy selection* refers to evaluating the utility of existing static preference pairs using the current model state π_{θ_t}, rather than generating new responses (online exploration). Our focus is on the efficient reweighting of the training corpus, avoiding the prohibitive computational cost of online generation while maintaining policy-adaptive supervision.

3.2 Coarse-Grained Candidate Generation

Evaluating the utility of all samples in $\mathcal{D}$ at every step is computationally intractable. To address this, SAGE uses a Refreshable Pool Strategy, which dynamically constructs candidate pools by sampling from difficulty-based subsets (easy, medium, hard) according to a time-dependent mixing ratio $\rho(k)$. This approach enables efficient sample selection by evolving the pool composition over time.

Difficulty-Based Partitioning. We initially partition $\mathcal{D}$ into three subsets $\{\mathcal{D}_{\text{easy}}, \mathcal{D}_{\text{medium}}, \mathcal{D}_{\text{hard}}\}$ based on off-policy difficulty priors. Each preference pair is annotated with two auxiliary judgments via an LLM-as-judge: (i) a *preference clarity* label $c_i \in \{\text{high}, \text{medium}, \text{low}\}$, indicating how clearly the chosen response is preferred over the rejected one, and (ii) a quality score $q_i \in \{1, \ldots, 5\}$ for the rejected answer (details in Appendix A.2). Based on the resulting annotations (c_i, q_i), we heuristically bucket samples into three difficulty strata:

$\mathcal{D}_{\text{easy}} = \{i \mid c_i = \text{high},\ q_i \leq 2\}$, $\mathcal{D}_{\text{medium}} = \{i \mid c_i = \text{medium},\ q_i \in \{2, 3\}\}$, and $\mathcal{D}_{\text{hard}}$ containing the remaining samples. These static difficulty priors are used only to provide a coarse stratification for candidate pool construction.

Dynamic Pool Construction. To approximate on-policy selection under limited computation, we divide training into K intervals, each associated with a refreshable candidate pool $\mathcal{P}_k$. Rather than treating the dataset as a static sequence, we partition $\mathcal{D}$ into K non-overlapping pools:

$$\mathcal{D} = \bigcup_{k=1}^{K} \mathcal{P}_k, \quad \text{s.t.} \quad \mathcal{P}_i \cap \mathcal{P}_j = \varnothing \tag{5}$$

where each pool contains $M = |\mathcal{D}|/K$ samples. Importantly, the composition of $\mathcal{P}_k$ is not uniform; it is constructed by sampling from difficulty subsets according to a time-dependent mixing ratio $\rho(k)$.

We define a linear schedule to control the evolution of the difficulty distribution:

$$\rho_c(k) = \rho_c^{\text{start}} + \left(\rho_c^{\text{end}} - \rho_c^{\text{start}}\right) \cdot \frac{k}{K} \tag{6}$$

where $c \in \{\text{easy}, \text{medium}, \text{hard}\}$ and $\sum_c \rho_c(k) = 1$. This schedule implements a coarse-grained curriculum, with higher weights on easier samples in early stages and a gradual shift toward harder samples in later stages. This piecewise-refresh strategy provides a practical compromise, preserving the essential dynamics of curriculum learning without the prohibitive cost of full-dataset re-evaluation at every step.

3.3 Fine-Grained SAGE Selection

Given the candidate pool $\mathcal{P}_k$, we apply a fine-grained filtering mechanism to improve the efficiency of policy updates. We propose the **SAGE Score**, a heuristic utility metric motivated by local properties of the loss function. By jointly considering gradient signal and prediction confidence, SAGE prioritizes informative samples while filtering out unstable ones, leading to more reliable policy updates.

Newton-Inspired SAGE Score. Standard data selection strategies typically prioritize samples with large gradient norms $\|\nabla_\theta \mathcal{L}\|$, implicitly assuming that larger gradients correspond to higher information content. However, such first-order criteria ignore the local curvature of the loss surface. In regions with high curvature, large gradients can lead to unstable updates and increased optimization variance.

To reason about the trade-off between gradient magnitude and update stability, we draw inspiration from the Newton decrement $\lambda(x)$, a classical second-order quantity that provides intuition about effective optimization progress:

$$\frac{1}{2}\lambda^2(x) \approx \frac{1}{2}\nabla_\theta \mathcal{L}(x)^\top \mathbf{H}^{-1}(x) \nabla_\theta \mathcal{L}(x), \tag{7}$$

where $\lambda^2(x)/2$ estimates the expected loss reduction under a Newton step. Directly computing the Hessian inverse $\mathbf{H}^{-1}$ is infeasible for large-scale models. Instead, we derive a tractable, sample-level approximation by exploiting the structure of the NCA objective (Eq. 4).

The NCA loss decomposes into a sum of response-level binary classification terms $\ell(r) = -\log \sigma(zr)$, where r denotes the model score for a response, $z \in \{+1, -1\}$ indicates whether the response is preferred or rejected, and $\sigma(\cdot)$ is the sigmoid function. Under this formulation, we approximate the per-sample Newton decrement by operating on these response-level score variables, ignoring cross-score correlations and adopting a diagonal Hessian approximation.

Let $p = \sigma(zr)$ denote the model confidence for a given response. For a single score variable, the diagonal Hessian entry corresponds to the curvature $h = \partial_r^2 \ell = p(1-p)$, while the squared gradient magnitude is $g^2 = (\partial_r \ell)^2 = (1-p)^2$. This yields a scalar approximation to the Newton decrement contribution for each response, decomposing the score into two computable proxies: a gradient-based signal term and a curvature-based stability term.

Reasoning trajectories can vary significantly in length, which may bias the score toward longer responses due to accumulated magnitudes. To mitigate this effect, we introduce a length-normalization term $L(x)$, defined as the total number of response tokens in the preference pair. Combining these components, we define the SAGE score $s_i(x)$ as a length-normalized, Newton-inspired measure of gradient efficiency:

$$\begin{aligned} s_i(x) &= \frac{1}{L(x)} \sum_{z \in \{y^w, y^l\}} \frac{g_z^2}{h_z + \epsilon} \\ &= \frac{1}{L_i} \sum_{z \in \{y^w, y^l\}} \frac{(1-p_z)^2}{p_z(1-p_z) + \epsilon}, \end{aligned} \tag{8}$$

Algorithm 1 SAGE: Stability-Aware Gradient Efficiency

Input: Dataset $\mathcal{D}$, policy π_θ, difficulty schedule $\rho(k)$, selection ratio schedule $\gamma(k)$.
Partition $\mathcal{D}$ into $\bigcup_{k=1}^{K} \mathcal{P}_k$ according to the difficulty schedule $\rho(k)$.
for $k = 1$ to K **do**
  // On-Policy Score Evaluation
  Perform a forward pass on $\mathcal{P}_k$ to compute length-normalized SAGE scores $\{s_i\}$ using Eq. 8.
  // Stability-Aware Filtering
  Compute the retention count $N_k = \gamma(k) \cdot |\mathcal{P}_k|$.
  Select the high-SNR subset $\mathcal{B}_k \leftarrow \{x_i \in \mathcal{P}_k \mid \text{rank}(s_i) \leq N_k\}$.
  // Policy Optimization
  Update the policy parameters $\theta \leftarrow \theta - \eta \nabla \mathcal{L}_{\text{NCA}}(\mathcal{B}_k)$.
end for

where ϵ is a Tikhonov damping term that ensures numerical stability in near-deterministic regimes.

Dynamic Batch Construction. After computing the SAGE score $s_i(x)$ for all candidates in the current pool $\mathcal{P}_k$, SAGE performs a hard filtering step to improve gradient efficiency. Specifically, samples are ranked in descending order of their scores, and only the top-γ_k fraction is retained:

$$\mathcal{B}_k = \{x_i \in \mathcal{P}_k \mid \text{rank}(s_i) \leq \gamma_k \cdot |\mathcal{P}_k|\}. \quad (9)$$

Here, γ_k is a time-dependent selection ratio. The policy is updated exclusively on the resulting high-SNR subset $\mathcal{B}_k$, while low-scoring samples are excluded from the backward pass.

Optimization-Theoretic Interpretation. The formulation in Eq. 8 and the associated hard truncation strategy differ from entropy-based active learning approaches. Instead, SAGE is motivated by two optimization principles that are particularly relevant for reasoning tasks.

- Curvature Regularization (Stability). The curvature proxy $h = p(1-p)$ reflects the sensitivity of the loss to changes in model confidence. Samples with high uncertainty ($p \approx 0.5$) often induce unstable gradients, which are particularly harmful for long chain-of-thought reasoning. By filtering out such samples, SAGE acts as a stability-aware filtering mechanism that mitigates high-variance updates.
- Signal-to-Noise Ratio (Efficiency). SAGE further seeks to improve the effective signal quality of the update direction. The score is large when the gradient signal g is strong while the curvature h is small, corresponding to *confident errors*. By physically removing samples with weak gradients ($g \to 0$) or excessive curvature ($h \gg 0$), SAGE ensures that retained gradients contribute meaningfully to loss reduction, leading to faster and more stable convergence than static or soft-weighted baselines.

4 Experiments

4.1 Experimental Setup

Model Selection. We use the Qwen2.5-Instruct family (Qwen et al., 2025) and evaluate three scales: 1.5B, 3B, and 7B. For 1.5B and 3B, we report the base Instruct checkpoints (their SFT variants underperform; see Appendix B.3). For 7B, we use the post-SFT checkpoint.

Training Data. We leverage the Light-R1 datasets (Wen et al., 2025) for training. SFT warm-up uses 76k samples from Stage 1.[1] For preference optimization, we construct **4,134** pairs by (i) deduplicating the original Light-R1 preference set[2] and (ii) adding additional pairs by generating rejected responses for randomly sampled Stage 1 instances. Details are provided in Appendix A.1.

Evaluation Benchmarks. To comprehensively evaluate the impact of our method on mathematical reasoning, we evaluate on a diverse suite of benchmarks, including GSM8K (Cobbe et al., 2021), MATH500, Minerva-MATH (Lewkowycz et al., 2022) (*Minerva*), Gaokao23-Math (Liao et al., 2024) (*Gaokao*), OlympiadBench (He et al., 2024) (*Olympiad*), CollegeMath (Tang et al., 2024) (*College*), AMC23, and AIME24. These benchmarks vary substantially in difficulty and scale, covering a wide range of problem types and reasoning depths. Additional statistics are provided in Appendix B.2.

Metrics. We report pass@8 on AIME24 and AMC23 due to their relatively small test set sizes and pass@1 on all other accuracy benchmarks. We set the maximum generation length to 10,000 tokens, which we found sufficient to reach perfor-

[1] https://huggingface.co/datasets/qihoo360/Light-R1-SFTData
[2] https://huggingface.co/datasets/qihoo360/Light-R1-DPOData

Method	GSM8K	MATH500	Minerva	Gaokao	Olympiad	College	AIME24	AMC23	Avg.
					Qwen2.5-1.5B-Instruct				
Vanilla	73.70	54.60	16.90	46.20	**22.70**	**38.40**	6.70	25.00	35.53
w/ DPO (Full)	74.70	56.20	19.50	47.30	20.00	38.00	**10.00**	22.50	36.03
w/ DPO (Random)	73.50	56.40	19.10	48.60	19.60	37.90	3.30	25.00	35.43
SAGE (Ours)	**74.80**	**57.20**	**20.20**	**50.40**	21.50	38.10	**10.00**	**27.50**	**37.46**
					Qwen2.5-3B-Instruct				
Vanilla	86.90	65.20	25.70	56.40	**27.70**	44.50	6.70	47.50	45.08
w/ DPO (Full)	86.40	65.60	27.20	56.90	27.00	44.90	10.00	50.00	46.00
w/ DPO (Random)	87.00	65.20	26.10	56.40	26.50	45.00	0.00	45.00	43.90
SAGE (Ours)	**87.50**	**66.00**	**28.30**	**58.23**	**27.70**	**45.14**	**13.30**	**55.00**	**47.65**
					Qwen2.5-7B-Instruct				
Vanilla	92.30	81.60	28.30	69.90	45.30	42.40	23.30	57.50	55.08
w/ DPO (Full)	92.70	82.00	29.40	70.60	**46.50**	42.70	26.70	62.50	56.64
w/ DPO (Random)	91.30	79.40	26.80	71.40	43.00	42.70	20.00	57.50	54.01
SAGE (Ours)	**93.10**	**82.80**	**33.10**	**71.40**	45.50	**43.10**	**33.30**	**70.00**	**59.04**

Table 1: Main results on mathematical reasoning benchmarks. *Vanilla* denotes the baseline models (either original Instruct or SFT variants as described in Model Selection). *DPO (Full)* and *DPO (Random)* indicate DPO training on the full preference dataset and a size-matched random subset, respectively. *Avg.* denotes the mean accuracy across benchmarks, and the best results are highlighted in **bold**.

mance saturation across datasets in preliminary experiments. Evaluation is performed using the SimpleRL-Reason framework[3].

Implementation. We build on 360-LLaMA-Factory[4] with sequence parallelism for long-CoT training. Unless specified otherwise, we use $\beta = 0.1$, 3 epochs, and batch size 16 on 8×A800 GPUs (sp=8) with a maximum sequence length of 32,768.

4.2 Main Results

Table 1 reports the performance of SAGE and baseline methods on mathematical reasoning benchmarks across three model scales. Across all model sizes (1.5B, 3B, and 7B), SAGE consistently outperforms standard DPO in average accuracy, indicating that its effectiveness is stable across capacity regimes. In particular, SAGE achieves the strongest gains at the 3B scale and reaches state-of-the-art performance across all eight benchmarks, suggesting improved utilization of preference supervision under constrained model capacity.

To disentangle the effect of data selection from data quantity, we further compare SAGE with a random DPO baseline using the same fraction of training data. As shown in Table 1, random subsampling consistently underperforms full DPO and often degrades performance relative to the vanilla models, indicating that naive data reduction can introduce optimization noise. In contrast, SAGE consistently improves over both full and random DPO, demonstrating that its gains stem from stability-aware and gradient-efficient data selection rather than reduced data volume. Moreover, the improvements are more pronounced on harder benchmarks and larger models, consistent with the motivation that stable optimization is critical for long CoT reasoning tasks.

Setting	GSM8K	MATH500	Minerva	Gaokao
SAGE	**74.80**	**57.20**	**20.20**	**50.40**
w/o s_1	73.70 (-1.10)	54.60 (-2.60)	16.90 (-3.30)	46.20 (-4.20)
w/o s_2	73.90 (-0.90)	55.30 (-1.90)	17.50 (-2.70)	47.10 (-3.30)
w/o g	74.00 (-0.80)	56.00 (-1.20)	18.00 (-2.20)	48.10 (-2.30)
w/o h	73.80 (-1.00)	55.40 (-1.80)	17.30 (-2.90)	47.50 (-2.90)

Table 2: Ablation results on Qwen2.5-1.5B-Instruct. w/o s_1, s_2, g, and h denote removing coarse-grained pool construction, fine-grained informativeness-based selection, gradient signal, and curvature regularization, respectively. Red values indicate absolute performance drops relative to full SAGE.

4.3 Ablation Study

Table 2 presents an ablation analysis of the key components in SAGE. Removing the coarse-grained difficulty-based sampling (s_1) results in consistent but modest performance drops, with larger degradation on harder benchmarks, reflecting its role in shaping appropriate candidate distributions. Removing the informativeness-based filtering (s_2) further degrades performance, show-

[3] https://github.com/hkust-nlp/simpleRL-reason

[4] https://github.com/Qihoo360/360-LLaMA-Factory

ing the importance of prioritizing samples with strong gradient signals. The largest performance drop occurs when curvature regularization is removed (w/o h), especially on challenging benchmarks such as Minerva and Gaokao, underscoring the role of stabilizing updates in high-curvature regions. Removing the gradient signal term (w/o g) also leads to noticeable degradation, indicating that gradient magnitude is essential for effective data selection. Overall, the results demonstrate that SAGE benefits from the complementary effects of gradient informativeness and curvature-aware stabilization, with difficulty-based sampling providing additional support on harder reasoning tasks.

4.4 Analysis

Unless otherwise specified, all analyses in this section are conducted on Qwen2.5-1.5B-Instruct and evaluated on the first four reasoning benchmarks.

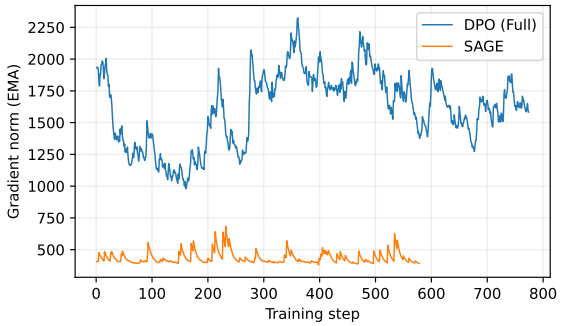


Figure 3: Gradient-norm dynamics during training.

Stability Analysis. Figure 3 compares the gradient-norm dynamics of standard DPO and SAGE during training. DPO exhibits large gradient magnitudes with frequent spikes across training, indicating high-variance and unstable updates. This behavior is especially pronounced in long chain-of-thought preference optimization, where ambiguous preference pairs can induce high-curvature regions of the loss landscape and amplify gradient noise.

In contrast, SAGE consistently reduces both the scale and variance of gradient norms from early training stages, resulting in smoother and more stable optimization trajectories. This pattern aligns with the design of SAGE, which incorporates curvature-aware scoring to filter high-curvature, low signal-to-noise samples and applies hard truncation to retain only informative gradients for backpropagation. Despite operating with smaller gradients, SAGE achieves stronger final performance,

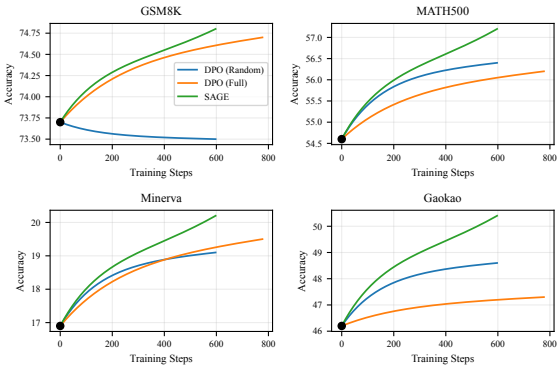


Figure 4: Training dynamics measured by evaluation performance over training steps.

suggesting that the improvement comes from more reliable and efficient optimization.

Training Dynamics. We analyze training behavior by tracking evaluation performance over training steps, with model checkpoints saved every 50 steps and evaluated independently. Due to different data budgets, the total number of training steps differs across methods: SAGE and DPO (Random) use the same number of preference pairs, while DPO (Full) runs longer. As shown in Figure 4, both DPO baselines exhibit early saturation, with limited improvement in later stages of training. In contrast, SAGE continues to improve steadily before gradually converging, indicating that its adaptive and stability-aware sample selection sustains learning progress over a longer period.

Computational Cost and Efficiency. We analyze the computational characteristics of SAGE in terms of effective training tokens and wall-clock training time. As shown in Figure 5(a), varying the keep ratio γ reveals a clear trade-off between accuracy and effective computation. Moderate values ($\gamma \in [0.4, 0.6]$) achieve higher accuracy than DPO (Full) while using substantially fewer effective training tokens, indicating improved gradient efficiency through selective backpropagation.

Figure 5(b) reports the corresponding wall-clock training time. Despite the reduction in effective training tokens, overall training time remains comparable across methods. This is because SAGE performs forward-only scoring on all candidate samples, while backpropagation is applied only to the retained subset; under a fixed memory budget, forward inference is substantially cheaper than gradient computation. As a result, training time is dominated by forward computation and system

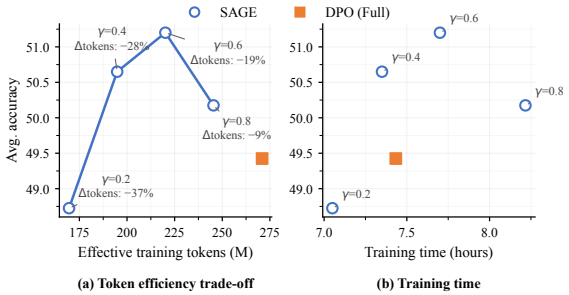


Figure 5: Accuracy versus effective training tokens (a) and wall-clock training time (b) under different ratios γ.

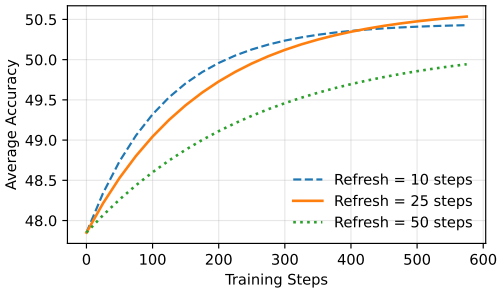


Figure 6: Sensitivity of SAGE to candidate-pool refresh frequency.

overheads rather than the number of backpropagated tokens. Overall, SAGE provides a flexible performance-efficiency trade-off without relying on unrealistic speedups.

Sensitivity to Pool Refresh Frequency. We analyze the sensitivity of SAGE to the candidate-pool refresh frequency by varying the refresh interval while keeping all other hyperparameters fixed. As shown in Figure 6, all configurations start from the same initial performance and improve consistently during training, indicating that SAGE is not sensitive to a specific refresh schedule. A moderate refresh interval (25 steps) achieves the best final performance, while both more frequent and sparser refreshing lead to slightly lower accuracy. This trend reflects a trade-off intrinsic to SAGE: overly sparse refreshing yields stale utility estimates under a changing policy, whereas overly frequent refreshing provides diminishing returns while increasing scoring overhead. Overall, SAGE exhibits a broad and stable operating region with respect to the pool refresh frequency.

5 Related Work

Preference Optimization for Math Reasoning. Reinforcement learning from human feedback (RLHF) has been widely used to improve mathematical reasoning in large language models. Early approaches relied on PPO (Schulman et al., 2017), but the need for a separate critic model incurs substantial computational overhead, especially for long chain-of-thought training. Recent methods such as ReMax (Li et al., 2023), GRPO (Shao et al., 2024), and DAPO (Yu et al., 2025) simplify optimization by removing the critic or stabilizing training dynamics. To avoid the complexity of online RL, Direct Preference Optimization (DPO) (Rafailov et al., 2023) and its variants, such as SimPO (Meng et al., 2024), CPO (Xu et al., 2024) and KTO (Ethayarajh et al., 2024), directly optimize over offline preference pairs. These methods primarily modify the training objective while typically assuming uniform utility across preference pairs, without explicitly addressing optimization stability under long chain-of-thought supervision (Morimura et al., 2024; Amini et al., 2024).

Data Selection and Optimization Stability. Recent work shows that training stability is strongly influenced by data selection (Zhou et al., 2024; Yu et al., 2024), especially for long chain-of-thought reasoning, where optimization often exhibits high gradient variance and ambiguous supervision (Cobbe et al., 2021; Arora and Zanette, 2025). Several approaches improve stability via static filtering or curriculum-based selection (Xia et al., 2024; Wang et al., 2024), but are typically applied off-policy with fixed criteria, limiting their ability to adapt to the model state. In contrast, SAGE performs on-policy, dynamic filtering over static preference data, adapting supervision to the current model state.

6 Conclusion

We propose SAGE, a stability-aware data selection framework for preference optimization in reasoning models. SAGE dynamically selects preference pairs by considering gradient informativeness and prediction confidence, leading to reliable optimization under long chain-of-thought supervision. Experiments across multiple model scales and mathematical reasoning benchmarks show that SAGE consistently outperforms standard DPO and random subsampling baselines, while improving gradient stability and data efficiency. Results highlight the benefits of on-policy, stability-aware supervision selection for aligning reasoning models.

Limitations

SAGE operates on a static preference corpus and does not consider online data collection or exploration. While this design keeps the method computationally efficient, performance depends on the coverage and quality of the preference data. In addition, SAGE introduces a forward-pass scoring step for estimating sample utility, which incurs extra overhead compared to static baselines, though this cost is mitigated by the pool-based design.

References

Afra Amini, Tim Vieira, and Ryan Cotterell. 2024. Direct preference optimization with an offset. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9954–9972.

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently. *arXiv preprint arXiv:2502.04463*.

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.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *Preprint*, arXiv:2402.14008.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Aitor Lewkowycz, Anders Johan Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Venkatesh Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems*.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. 2023. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv preprint arXiv:2310.10505*.

Minpeng Liao, Chengxi Li, Wei Luo, Wu Jing, and Kai Fan. 2024. MARIO: MAth reasoning with code interpreter output - a reproducible pipeline. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 905–924, Bangkok, Thailand. Association for Computational Linguistics.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, and 1 others. 2025. Deepscaler: Surpassing o1-preview with a 1.5 b model by scaling rl. *Notion Blog*.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Tetsuro Morimura, Mitsuki Sakamoto, Yuu Jinnai, Kenshi Abe, and Kaito Ariu. 2024. Filtered direct preference optimization. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22729–22770.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

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

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. *Preprint*, arXiv:2403.02884.

Jiachen Tianhao Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. 2024. Greats: Online selection of high-quality data for llm training in every iteration. *Advances in Neural Information Processing Systems*, 37:131197–131223.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Tanglifu Tanglifu, Xiaowei Lv, and 1 others. 2025. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 318–327.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 54104–54132.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. In *International Conference on Machine Learning*, pages 55204–55224. PMLR.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

Zichun Yu, Spandan Das, and Chenyan Xiong. 2024. Mates: Model-aware data selection for efficient pretraining with data influence models. *Advances in Neural Information Processing Systems*, 37:108735–108759.

Kun Zhou, Beichen Zhang, Zhipeng Chen, Xin Zhao, Jing Sha, Zhichao Sheng, Shijin Wang, Ji-Rong Wen, and 1 others. 2024. Jiuzhang3. 0: Efficiently improving mathematical reasoning by training small data synthesis models. *Advances in Neural Information Processing Systems*, 37:1854–1889.

A Methodology

A.1 Data Construction

The original Light-R1 DPO dataset contains **3,000** samples. Preliminary analysis revealed substantial noise, mainly from redundant queries and *degenerate preference pairs* in which the chosen and rejected responses were identical. To improve the reliability of preference learning, we applied a two-stage deduplication pipeline:

- **Duplicate Query Filtering**: We removed redundant queries to reduce the risk of overfitting to specific prompts.
- **Preference Consistency Check**: We discarded pairs whose final answers (extracted from the `\boxed{}` markers) were identical between the chosen and rejected responses.

After filtering, we obtained a high-quality subset of **1,659** samples. To restore the dataset scale, we sampled **5,000** unique instances from the SFT data (as detailed in Section 4.1), ensuring zero overlap with the retained samples. To generate rejected responses, we used *DeepScaleR-1.5B-Preview*[5] (Luo et al., 2025), a small model fine-tuned specifically for reasoning tasks. By using this model, we ensured that the augmented dataset maintained the same distribution and characteristics as the original, providing consistency in the data generation process. The prompt we employed for generating rejected responses was:

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

To maintain the validity of the DPO triplets, we filtered out any generated response whose final answer matched the ground-truth answer. This process produced **2,475** valid negative pairs from the 5,000 instances. Combined with the retained subset, the final dataset contains **4,134** samples.

A.2 Preference Annotation

To implement difficulty-based partitioning for SAGE, we used *DeepSeek-V3* (Liu et al., 2024) as a judge to provide multi-dimensional annotations for each triplet (x_i, y_i^w, y_i^l). The annotation process evaluates two dimensions to assess the optimization utility of each pair:

- **Preference Clarity**: c_i measures the degree of distinctness between the chosen and rejected solutions [High, Medium, or Low].
- **Rejected Response Quality**: q_i is an absolute quality score (1–5) assessing the logical coherence and depth of the rejected response.

We use this dual-dimension approach because the utility of a preference pair depends on the plausibility of the error. A pair exhibits *High Clarity* if the rejected solution is obviously flawed, providing a strong signal for early alignment. Conversely, *Low Clarity* samples, where the rejected response follows a sophisticated but flawed reasoning path, are crucial for fine-tuning the boundaries in later training stages. The judge model uses the following structured template:

System Prompt
You are an expert mathematical evaluator. You will be presented with a math problem and two model-generated solutions: ***Chosen*** *(correct answer) and* ***Rejected*** *(incorrect answer).*

Task 1: Preference Clarity Assessment
Evaluate the distinctness between the solutions:
- ***High****: The rejected solution is obviously flawed; the preference is stark and easy to identify.*
- ***Medium****: The rejected solution is plausible but contains noticeable errors.*
- ***Low****: The rejected solution is highly deceptive and resembles the chosen one, requiring deep analysis to refute.*

Task 2: Rejected Solution Quality Analysis
Rate the quality of the ***Rejected*** *reasoning path (1–5) based on its coherence and depth, despite its incorrect conclusion.*

Format*: Return output strictly in the following JSON format:*

```
{
  "clarity": "High | Medium | Low",
  "reason": "...",
  "rejected_analysis": "...",
  "rejected_score": x
}
```

Inputs
[Problem]*: {{problem_text}}*
[Chosen]*: {{chosen_solution}}*
[Rejected]*: {{rejected_solution}}*

Analysis of Data Distribution The joint distribution is summarized in Table 3. Consistent with the objectives of SAGE, most samples (3,018) fall into the *High Clarity* category, providing a robust starting point for curriculum learning. The scarcity in the *Low Clarity* and *High Quality* regions (e.g., Q5) is expected, as highly sophisticated yet incorrect reasoning paths are rare.

[5] https://huggingface.co/agentica-org/DeepScaleR-1.5B-Preview

Clarity	Q1	Q2	Q3	Q4	Q5	Total
High	946	1,728	221	124	2	3,018
Medium	0	572	474	11	0	1,057
Low	0	39	11	2	5	59
Total	946	2,339	706	139	7	**4,134**

Table 3: Joint distribution of Preference Clarity and Rejected Response Quality (Q1–Q5). High Clarity indicates obvious errors.

Curriculum Subset Results Based on the partitioning principles in Section 3.2, we categorized the **4,134** triplets into three pools:

- **Easy Pool** ($\mathcal{D}_{\text{easy}}$): High Clarity and $q_i \leq 2$ (2,674 samples).
- **Medium Pool** ($\mathcal{D}_{\text{medium}}$): Medium Clarity and $q_i \in \{2, 3\}$ (1,046 samples).
- **Hard Pool** ($\mathcal{D}_{\text{hard}}$): Remaining Low Clarity or High Quality samples (414 samples).

B Experiments

B.1 Training Details

The training hyperparameters and SAGE scheduling configurations are summarized in Table 4.

Hyperparameter	Value
Preference loss coefficient (β)	0.1
Training epochs	3
Batch size	16
Sequence parallelism (sp)	8
Maximum sequence length	32,768
Learning rate	5×10^{-7}
Difficulty mix ρ_{start}	easy: 0.90 medium: 0.10 hard: 0.00
Difficulty mix ρ_{end}	easy: 0.40 medium: 0.40 hard: 0.20
Keep ratio γ_{start}	1.0
Keep ratio γ_{end}	0.4
refresh step	25

Table 4: Training hyperparameters and SAGE scheduling configurations.

Difficulty Mixing Schedule. The difficulty mixing ratios are chosen empirically based on the dataset distribution after difficulty partitioning. Specifically, the easy, medium, and hard subsets contain 2,674, 1,046, and 414 samples, respectively. We therefore initialize training with a higher proportion of easy samples to provide stable early supervision, and gradually increase the weights of medium and hard samples through linear annealing. This schedule balances early optimization stability with sufficient exposure to harder preference pairs in later training stages.

B.2 Dataset

See Table 5 for statistics of test sets.

Dataset	# Problems
GSM8K	1,319
MATH500	500
Minerva	272
Gaokao	395
Olympiad	675
CollegeMath	2,818
AIME24	30
AMC23	40

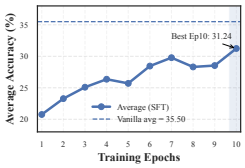
Table 5: Statistics of the test sets used for evaluation.

B.3 SFT Results

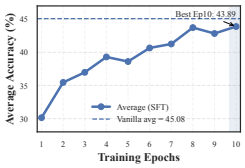
We use the Light-R1 Stage 1 dataset to fine-tune the 1.5B-, 3B-, and 7B-Instruct models. The training hyperparameters follow those of Light-R1: the cutoff length is set to 20,000 tokens, the batch size is 128, the learning rate is 5×10^{-6}, and training is conducted for 10 epochs. We evaluate checkpoints from all epochs on 8 benchmarks and select the final checkpoint based on the average accuracy across benchmarks.

1.5B and 3B. Figure 7 presents the SFT results for the 1.5B and 3B models over 10 training epochs. For both model sizes, the average accuracy increases steadily as training proceeds, indicating that supervised fine-tuning effectively improves model performance. However, none of the SFT checkpoints within the 10 epochs surpass the corresponding vanilla baselines. Even at their best-performing epochs, the SFT models remain below the original Qwen2.5-Instruct models in terms of average accuracy across benchmarks.

This observation suggests that, at smaller model scales, SFT alone may be insufficient to fully exploit the diverse reasoning patterns in the Light-R1 data. Consequently, we directly adopt the original Qwen2.5-1.5B-Instruct and Qwen2.5-3B-Instruct



(a) Qwen2.5-1.5B-Instruct



(b) Qwen2.5-3B-Instruct

Figure 7: SFT results for small-scale models (1.5B and 3B). Although performance improves over training epochs, none of the checkpoints surpass the vanilla baselines.

models as the vanilla baselines for subsequent experiments.

7B. The SFT results for the 7B model are shown in Figure 8. Checkpoints from epochs 5 to 10 consistently achieve higher average accuracy than the vanilla baseline, with epoch 5 yielding the best overall performance. Accordingly, we select the epoch-5 checkpoint as the vanilla model to replace Qwen2.5-7B-Instruct in subsequent experiments.

Despite the overall improvement, the SFT model underperforms the original Qwen2.5-7B-Instruct model on **Minerva Math** and **College Math**. Specifically, Qwen2.5-7B-Instruct achieves accuracies of 37.10 and 46.50 on these benchmarks, whereas the selected SFT checkpoint (epoch 5) attains 28.30 and 42.40, respectively. Within the SFT training process, the best performance on Minerva Math and College Math is achieved at epoch 10 (29.40) and epoch 9 (43.20), respectively, indicating that these benchmarks may benefit from later-stage fine-tuning.

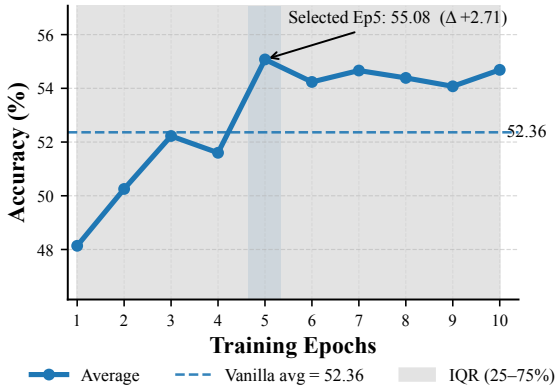


Figure 8: SFT results for the Qwen2.5-7B-Instruct model across training epochs.