MECHDS: MECHANISTIC RLVR DATA SELECTION VIA REASONING COMPONENT DEPENDENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) has become a central paradigm for improving the reasoning capabilities of large language models, yet effective data selection remains a key challenge. Existing RLVR data selection methods typically estimate the value of training data through extrinsic proxy signals, while the model's internal reasoning mechanisms remain largely unexplored. We study this missing dimension through reasoning component dependence (RCD), which measures how strongly a training sample relies on components critical to reasoning during generation. To examine information beyond extrinsic proxies, we compare RLVR training on samples with similar behavioral entropy but different reasoning component dependence. Empirically, a consistent pattern emerges across entropy-matched comparisons: low-dependence samples yield better RLVR performance than high-dependence samples, while high-dependence samples converge faster and exhibit smaller answer–reasoning gaps. These findings motivate **MechDS**, a dynamic data selection strategy that adaptively balances stability, learning potential, and exploration according to behavioral entropy and reasoning component dependence. Experiments on three models and three benchmarks show that MechDS consistently improves RLVR performance over strong data selection baselines, demonstrating that reasoning component dependence provides mechanistic information beyond extrinsic signals for selecting effective RLVR data.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (Lambert et al., 2024; Guo et al., 2025; Yu et al., 2026) (RLVR) has emerged as an effective paradigm for improving the reasoning capabilities of large language models. By optimizing policies against automatically checkable outcomes, RLVR can substantially improve mathematical and logical reasoning without requiring dense human supervision. Its effectiveness, however, depends strongly on the training data. Applying RLVR to all available prompts is computationally expensive, while indiscriminately including low-value samples may dilute useful learning signals or encourage undesirable optimization behavior. Selecting a compact set of effective training samples is therefore a central problem for scalable RLVR.

Existing RLVR data selection methods estimate training value through various *extrinsic proxy signals*. These include sample-level behavior or difficulty (Laurençon et al., 2022; Ankner et al., 2025; Li et al., 2024; Xia et al., 2024; Li et al., 2025b), subset-level properties such as coverage, diversity, and exploration potential (Tang et al., 2025), and optimization-level feedback such as gradient or learning-trajectory alignment (Li et al., 2025a; Yang et al., 2026). Despite their different formulations, these signals characterize training data through observable statistics, set structure, or optimization dynamics, while leaving the model's internal reasoning mechanisms largely unexplored. Consequently, samples with similar extrinsic proxies may appear equally valuable despite relying on substantially different internal computations.

Motivated by this, we study this missing dimension through *reasoning component dependence* (RCD), a mechanistic signal that measures how strongly a sample relies on components critical to reasoning during generation. To examine whether RCD provides information beyond existing extrinsic proxy signals, we jointly analyze RLVR data through behavioral entropy (BE) and RCD, comparing samples with similar entropy but different component dependence. Evaluation considers

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

not only final-answer performance, measured by Mean@16, but also reasoning validity, assessed using an LLM-as-Judge. The Judge determines whether a correct answer is supported by a logically sound reasoning process rather than reached through flawed or unsupported reasoning. The difference between Mean@16 and the Judge score is defined as the answer–reasoning gap. A consistent pattern emerges: low-RCD samples yield stronger final performance, whereas high-RCD samples reach target performance faster and exhibit a smaller answer–reasoning gap. Further analysis of component importance suggests that high-RCD training largely preserves established internal computations, while low-RCD training induces broader changes and offers greater learning potential. Based on these findings, we propose **MechDS**, an interpretable data selection method that adapts across training stages to balance stability, learning potential, and exploration in the BE–RCD space.

We evaluate MechDS on three reasoning models, Qwen2.5-Math-7B-Base, Qwen2.5-Math-1.5B-Instruct (Yang et al., 2024), and DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025), across three benchmarks. MechDS consistently outperforms strong RLVR data selection baselines. Our contributions are summarized as follows:

- We introduce *reasoning component dependence*, a mechanistic signal that measures how strongly an RLVR sample relies on components critical to reasoning during generation.
- Experiments reveal complementary training roles across the BE–RCD space: high-RCD samples enable faster optimization with smaller answer–reasoning gaps, whereas low-RCD samples provide larger final performance gains through broader internal adaptation.
- We propose **MechDS**, an RLVR data selection method that adapts across training stages to balance stability, learning potential, and exploration, consistently outperforming data selection baselines.

2 Related Work

Data Selection for RLVR Existing RLVR data selection methods estimate training value through several types of extrinsic proxy signals. Some methods assign each instance a scalar utility based on sample-level behavior or difficulty and retain high-scoring samples (Laurençon et al., 2022; Ankner et al., 2025; Li et al., 2024; Xia et al., 2024; Li et al., 2025b). Other methods improve data efficiency by constructing subsets with better coverage, diversity, or exploration potential (Tang et al., 2025). More recent work selects samples using optimization-level feedback, such as gradient alignment or learning trajectory alignment (Li et al., 2025a; Yang et al., 2026). While these methods have improved RLVR data efficiency, their selection signals are derived from sample statistics, verifier-induced behavior, set-level structure, or optimization dynamics. In contrast, we study data selection through reasoning component dependence, asking whether a sample's reliance on components critical to reasoning during generation provides information beyond these extrinsic proxies.

Circuit-Based Mechanistic Interpretability. Mechanistic interpretability aims to uncover mechanisms inside neural networks that humans can understand (Olah et al., 2020; Vig & Belinkov, 2019). Within this paradigm, circuit analysis studies how sparse groups of attention heads, MLP neurons, or features implement behaviors such as induction, copying, composition, and information routing (Elhage et al., 2021; Ameisen et al., 2025; Lindner et al., 2023). Causal interventions, including activation patching, attention head ablation, and attribution tracing, connect these components to model behavior by perturbing or tracing information flow through the network (Ameisen et al., 2025). Recent work extends this perspective to reasoning, showing that complex reasoning behavior can be associated with causally important model components (Kim et al., 2025; Hong et al., 2026). Closely related to our work, CircuitSeer (Wang et al., 2025) uses circuits identified from mathematical reasoning data to guide data selection for supervised fine-tuning. Inspired by this line of work, we identify components relevant to reasoning through causal perturbation on valid reasoning trajectories and study whether their dependence provides useful signals for RLVR data selection.

3 Data Signals for RLVR Selection

This section introduces the two signals used in our empirical study of RLVR data selection. The first is a mechanistic signal, *reasoning component dependence*, which measures how strongly a training sample relies on components critical to reasoning during generation. The second is a behavioral signal, *behavioral entropy*, which summarizes the model's output uncertainty.

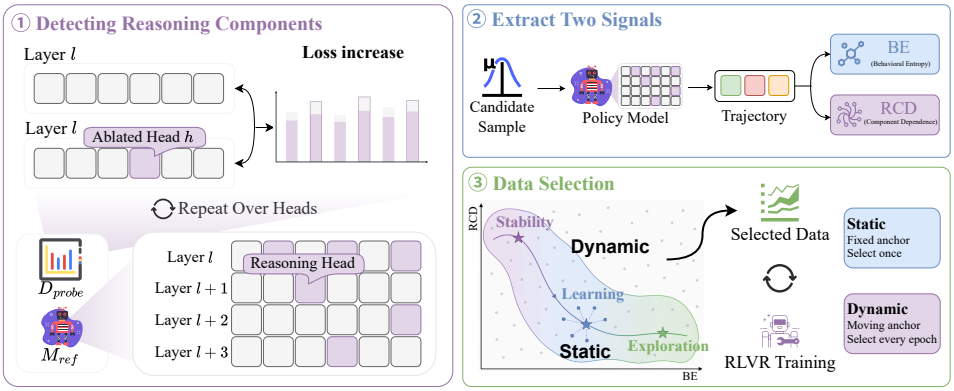

Figure 1: Overview of MechDS, which identifies components critical to reasoning, extracts BE and RCD signals, and performs stage-adaptive data selection for RLVR training.

3.1 Mechanistic Signal: Reasoning Component Dependence

3.1.1 Identifying Components Critical to Reasoning

Prior work in circuit analysis suggests that Transformer computations can often be traced to sparse circuits formed by attention heads with distinct functional roles (Elhage et al., 2021; Ameisen et al., 2025; Wang et al., 2025). Motivated by this view, we identify attention heads critical to reasoning through causal ablation on a probe set, and use the resulting set as the reference component set for measuring sample dependence.

Probe construction. To identify components critical to reasoning, we construct the probe set from trajectories that contain both correct outcomes and valid reasoning processes. Given a candidate problem set $\mathcal{D} = \{x_i\}_{i=1}^{N}$, we first use the target model, which is also the model to be trained, to generate a solution trajectory y_i for each problem x_i. We then apply two filters: an answer verifier $V(\cdot)$ checks whether the final answer is correct, and an LLM-based reasoning judge $J(\cdot)$ checks whether the reasoning process is valid. This yields a set of valid reasoning trajectories:

$$\mathcal{D}_{\text{valid}} = \{(x_i, y_i) \mid V(x_i, y_i) = 1,\ J(x_i, y_i) = 1\}. \tag{1}$$

To focus the probe set on trajectories that require nontrivial reasoning by the target model, we further rank $\mathcal{D}_{\text{valid}}$ by the target model's path entropy. For a trajectory (x, y), we define

$$H_\theta(x, y) = \frac{1}{|y|} \sum_{t=1}^{|y|} H(p_\theta(\cdot \mid x, y_{<t}))\,. \tag{2}$$

where $p_\theta(\cdot \mid x, y_{<t})$ is the target model's next-token distribution. We select the top-K_{prob} trajectories with the largest $H_\theta(x, y)$ as the probe set. This construction retains only valid reasoning trajectories while emphasizing samples on which the target model exhibits greater reasoning uncertainty.

Reasoning component identification. Let $\mathcal{M}_{\text{ref}}$ denote the reference model used for component identification, $\mathcal{D}_{\text{probe}}$ the curated probe set for evaluating component importance, and $\mathcal{H}$ the set of all attention heads in $\mathcal{M}_{\text{ref}}$. For each candidate head $h \in \mathcal{H}$, we ablate the head with undifferentiated attention (Zhou et al., 2025), replacing its attention pattern with a causal uniform distribution while keeping its value pathway unchanged. This intervention preserves the causal structure and output scale of the head, but removes its specific information routing pattern.

Let $\mathcal{M}_{\text{ref}}^{-h}$ denote the model after ablating head h, and let $\mathcal{L}(x, y; \mathcal{M})$ denote the language modeling loss of model $\mathcal{M}$ on trajectory (x, y). We define the importance score of head h as the expected loss increase on the probe set:

$$S(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{probe}}} \left[\mathcal{L}(x, y; \mathcal{M}_{\text{ref}}^{-h}) - \mathcal{L}(x, y; \mathcal{M}_{\text{ref}})\right]. \tag{3}$$

A larger $S(h)$ indicates that head h is more important for valid reasoning trajectories with high entropy. We sort all attention heads by $S(h)$ in descending order and select the top-K_{head} heads as the reference component set $\mathcal{H}^\star$.

3.1.2 Measuring Reasoning Component Dependence

Given the reference component set $\mathcal{H}^\star$, sample dependence is measured by comparing the original and ablated models along the same generated trajectory. For each problem x, a solution trajectory $y \sim \pi_\theta(\cdot \mid x)$ is first generated by the target policy model. We then evaluate the same trajectory under two conditions: the original model, with next token distribution $p_\theta(\cdot \mid x, y_{<t})$, and an ablated model where all heads in $\mathcal{H}^\star$ are replaced by undifferentiated attention (Zhou et al., 2025). Let $p_\theta^{-\mathcal{H}^\star}(\cdot \mid x, y_{<t})$ denote the ablated next token distribution. The reasoning component dependence score is defined as:

$$\mathrm{RCD}(x, y) = \frac{1}{|y|} \sum_{t=1}^{|y|} D_{\mathrm{KL}} \left(p_\theta(\cdot \mid x, y_{<t}) \,\|\, p_\theta^{-\mathcal{H}^\star}(\cdot \mid x, y_{<t}) \right). \tag{4}$$

A high RCD indicates that ablating the reference components substantially changes the model's predictions throughout the generated trajectory, reflecting strong dependence on components identified as important for valid reasoning trajectories. Conversely, a low RCD indicates weaker dependence.

3.2 Behavioral Signal: Behavioral Entropy

In addition to reasoning component dependence, we use behavioral entropy as an external signal of the target policy's uncertainty on each candidate training sample. For consistency with the RCD computation, behavioral entropy is computed on the same trajectory generated by the target policy model. Given a problem x and its generated solution trajectory $y \sim \pi_\theta(\cdot \mid x)$, we define the behavioral entropy of (x, y) as the average next token entropy along the trajectory:

$$\mathrm{BE}(x, y) = \frac{1}{|y|} \sum_{t=1}^{|y|} H(p_\theta(\cdot \mid x, y_{<t})). \tag{5}$$

Behavioral entropy (BE) captures the model's uncertainty during generation, providing a natural proxy for sample difficulty or exploration potential. We use BE together with RCD in the following analysis and data selection.

4 A Two-Axis View of RLVR Training Data

Behavioral entropy tells us how uncertain the target policy is, but not what internal computation supports its generation. Two samples may look equally difficult under entropy while differing substantially in their reliance on components critical to reasoning. We therefore analyze RLVR training data through two axes: behavioral entropy as an external uncertainty signal, and reasoning component dependence as a mechanistic signal. To characterize their training effects, we evaluate final answer accuracy using Mean@16, computed over 16 sampled responses per problem, and reasoning validity using an LLM-as-Judge. Their difference is referred to as the answer–reasoning gap.

4.1 Analysis Setup

We conduct this analysis on the DAPO-Math training set, using Qwen2.5-Math-7B-Base (Yang et al., 2024) as the backbone model. Following Section 3, we compute behavioral entropy $\mathrm{BE}(x, y)$ and reasoning component dependence $\mathrm{RCD}(x, y)$ for each candidate sample. We partition candidate samples into three entropy bands according to their normalized $\mathrm{BE}(x, y)$ scores: $[0, 0.2]$ as the low-entropy band, $[0.4, 0.6]$ as the mid-entropy band, and $[0.8, 1.0]$ as the high-entropy band. Within each entropy band, we further rank samples by $\mathrm{RCD}(x, y)$ and select high-RCD and low-RCD subsets. This design yields entropy-matched comparisons where samples have similar behavioral uncertainty but different reasoning component dependence. Each subset contains 1024 samples, ensuring that all comparisons are matched in data scale. We train each subset with GRPO (Guo et al., 2025)

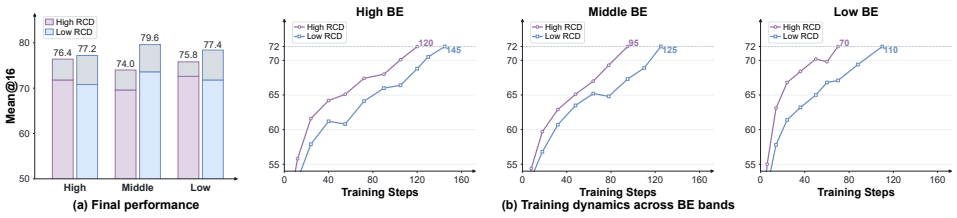

Figure 2: Training effects across BE–RCD regions. (a) Final performance measured by Mean@16 and LLM-as-Judge scores. (b) Training dynamics across BE bands, showing Mean@16 over training steps.

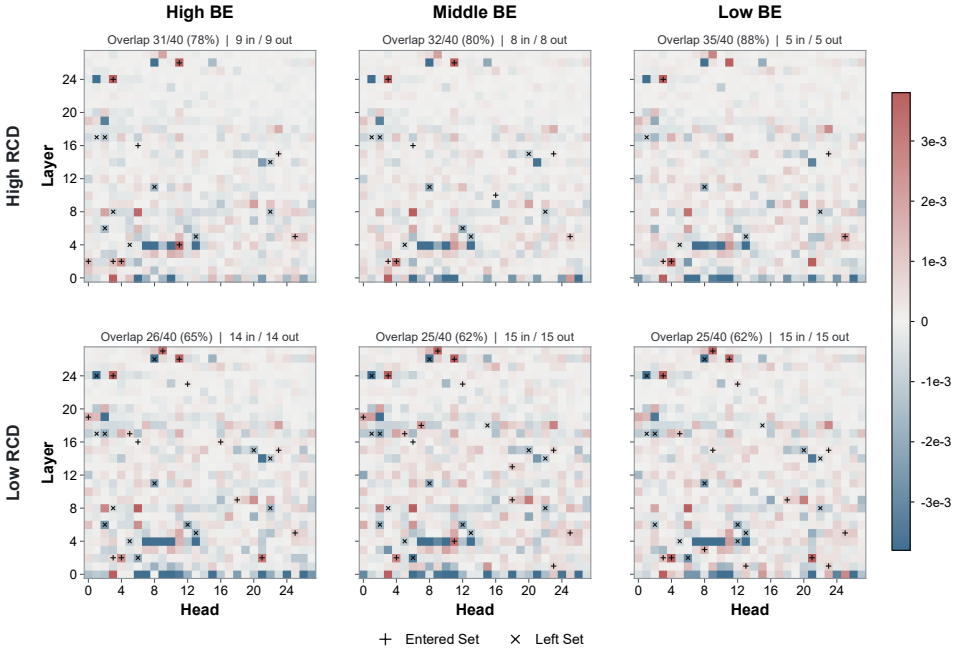

Figure 3: Head importance changes after RLVR across six BE–RCD regions, computed as trained minus base. "+" and "×" mark heads entering and leaving the top-40 reference component set.

implemented in the Verl (Sheng et al., 2025) framework under the same training configuration. We evaluate model outputs on MATH-500 (Hendrycks et al., 2021) from two perspectives: final-answer correctness measured by Mean@16, and reasoning-process validity judged by DeepSeek V4-Flash (Xu et al., 2026) as an LLM-as-Judge. Each experiment uses three random seeds, and we report mean performance.

4.2 Training Effects Across BE–RCD Regions

Low-RCD samples yield stronger final performance beyond extrinsic proxies. As shown in Figure 2(a), low-RCD subsets consistently outperform their high-RCD counterparts within the same BE band, with the mid-BE low-RCD subset achieving the best overall Mean@16. This advantage cannot be explained by common extrinsic proxy signals. Figure 4 shows that high-RCD subsets obtain higher rewards in the low- and high-BE bands and comparable rewards in the mid-BE band, while the relative response length reverses across BE bands. Despite these inconsistent proxy patterns, low-RCD subsets consistently achieve better final performance, indicating that RCD captures training-relevant mechanistic information beyond extrinsic proxy signals.

High-RCD samples reach target performance faster and exhibit a smaller answer–reasoning gap. As shown in Figure 2(b), high-RCD subsets reach the performance threshold with fewer

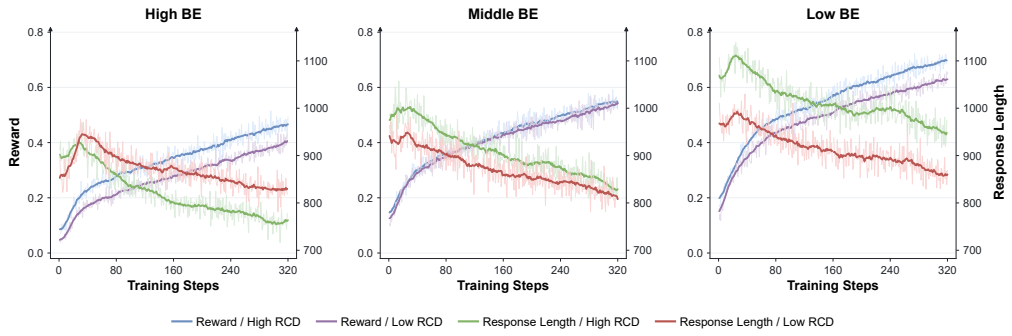

Figure 4: Training reward and response-length dynamics across BE bands.

training steps within every BE band, with the low-BE high-RCD subset being the fastest. Figure 2(a) shows that high-RCD subsets consistently exhibit a smaller gap between Mean@16 and the LLM-as-Judge score, with the low-BE high-RCD subset achieving the smallest gap. Thus, although high-RCD samples provide lower Mean@16, they enable faster and more reasoning-consistent optimization.

Component evolution clarifies the distinct optimization roles of different BE–RCD regions. Figure 3 reveals a clear stability–plasticity trade-off across RCD regions. High-RCD samples depend more strongly on the identified components, and training on these samples largely preserves the original reference component set. This suggests that optimization mainly refines existing computation patterns rather than substantially reorganizing them. Consistent with this interpretation, high-RCD subsets reach the target performance with fewer training steps and exhibit smaller answer–reasoning gaps. However, remaining close to these established patterns may also constrain further adaptation, which is consistent with the lower final performance of high-RCD training. Low-RCD samples, in contrast, depend less on the identified components and induce broader internal reconfiguration, causing larger changes to the model's existing computation patterns. Such exploration may succeed or fail during optimization, leading to slower attainment of target performance and larger answer–reasoning gaps. At the same time, it creates more opportunities to discover more effective computation patterns, providing a plausible explanation for the higher final performance of low-RCD samples.

5 MechDS

Section 4.2 shows that different BE–RCD regions play complementary roles in RLVR training: low-BE high-RCD samples stabilize early optimization, mid-BE low-RCD samples provide the main performance gains, and high-BE low-RCD samples support late-stage exploration. MechDS translates these findings into a stage-adaptive data selection strategy over the BE–RCD space.

For each candidate sample i, we compute BE and RCD once before training and apply min–max normalization over the candidate set, yielding $b_i, r_i \in [0, 1]$. For an E-epoch training run, we define the normalized progress at epoch e as $\tau_e = (e-1)/(E-1)$, where $e \in \{1, \ldots, E\}$. MechDS uses a moving anchor to represent the preferred BE–RCD region at each training stage. Let $\alpha \in (0, 0.5]$, and define the early- and late-stage progress as

$$\rho(\tau) = \min\left(\frac{\tau}{\alpha}, 1\right), \qquad \gamma(\tau) = \max\left(\frac{\tau - (1-\alpha)}{\alpha}, 0\right). \tag{6}$$

The stage-dependent anchor is then defined as

$$\bar{b}(\tau) = \frac{1}{2}\rho(\tau) + \left(\beta - \frac{1}{2}\right)\gamma(\tau), \qquad \bar{r}(\tau) = 1 - \rho(\tau), \tag{7}$$

where $\beta \in [0.5, 1]$ denotes the terminal BE target. The anchor starts from $(0, 1)$, reaches $(0.5, 0)$ after the first α fraction of training, and remains there during the middle stage. During the final α fraction, it moves toward $(\beta, 0)$. Thus, α controls the duration of the transition stages, while β

controls the strength of late-stage exploration. After constructing the anchor trajectory, we measure the squared distance between sample i and the current anchor:

$$d_i(\tau) = \left(b_i - \bar{b}(\tau)\right)^2 + \left(r_i - \bar{r}(\tau)\right)^2. \tag{8}$$

At the beginning of each epoch, MechDS selects K samples with the smallest $d_i(\tau_e)$ for training.

6 Main Experiments

6.1 Experimental Setup

We evaluate MechDS on Qwen2.5-Math-7B, Qwen2.5-Math-1.5B-Instruct, and DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025) on MATH-500, AIME 2024, and AIME 2025, using the same DAPO-Math candidate pool and GRPO training framework as in Section 4.1. All methods use a 20% selection budget, applied at each epoch for dynamic methods. Baselines include Random, Random-Dynamic, PPL-Top (Laurençon et al., 2022), PPL-Middle (Ankner et al., 2025), CircuitSeer (Wang et al., 2025), LIMR (Li et al., 2025b), LearnAlign (Li et al., 2025a), and DEPO (Tang et al., 2025). Random-Dynamic uniformly re-samples the training subset at each epoch, while DEPO also performs dynamic data selection during RLVR training. MechDS-Static selects once using the fixed anchor $(0.5, 0)$, whereas MechDS-Dynamic re-ranks the pool each epoch along the stage-dependent anchor trajectory, with $\alpha = 0.25$ and $\beta = 0.75$ by default. Each experiment uses three random seeds, and we report mean performance. Full configurations are provided in Appendix A.

6.2 Main Results

Table 1 reports the main results across the three models. MechDS-Dynamic consistently achieves the best Mean@16 and Judge scores across all settings. Importantly, MechDS-Static also surpasses all competing baselines, including DEPO, which performs dynamic data selection during RLVR training. Since MechDS-Static uses a fixed subset throughout training, these results show that combining BE and RCD already provides an effective selection signal without dynamic re-selection. MechDS-Dynamic further improves over the static variant, demonstrating the additional benefit of adapting data composition as training progresses. The advantage is also evident in reasoning validity. MechDS-Dynamic consistently achieves higher Judge scores and smaller answer–reasoning gaps across all three models. The improvement in Judge is larger than that in Mean@16 for every model, suggesting that MechDS not only improves final answer accuracy but also reduces correct answers supported by invalid or unsupported reasoning. These results support the value of RCD as a complementary selection signal and the benefit of adapting data composition throughout RLVR training.

7 Analysis and Discussion

7.1 Reasoning Component Analysis

Component controls. We construct two control variants by replacing the identified components while keeping all other settings unchanged. **Random Heads** uses 40 randomly sampled attention heads, whereas **Low Heads** uses the 40 heads with the lowest $S(h)$ scores. These two controls test whether RCD merely benefits from perturbing a fixed number of attention heads, or specifically depends on the heads identified by $S(h)$ as important for valid reasoning trajectories. As shown at the bottom of Table 2, both controls substantially underperform MechDS-Dynamic, with Low Heads showing the largest degradation. The clear gap between the identified components and these control sets indicates that the effectiveness of RCD is tied to the specific components selected by $S(h)$ rather than to generic attention head perturbation. Moreover, the further degradation observed with Low Heads supports the informativeness of the head ranking score itself.

Component size sensitivity. We further study the sensitivity to the number of selected components, denoted by K_{head}. We vary $K_{\text{head}} \in \{20, 40, 60, 80\}$ while keeping the remaining settings fixed, with $K_{\text{head}} = 40$ as the default. As shown in Figure 5(a), performance first improves as K_{head} increases from 20 to 40, remains competitive at 60, and then declines at 80. The best Mean@16 of

Table 1: Main results on three reasoning benchmarks. Judge is the reasoning validity score, followed by $\downarrow \Delta = \text{Mean@16} - \text{Judge}$. * denotes dynamic data selection during RLVR.

Model	Method	AIME24		AIME25		MATH-500		Average	
		Mean@16	Judge	Mean@16	Judge	Mean@16	Judge	Mean@16	Judge
Qwen2.5-Math-7B	Random	23.75	17.92 ↓5.83	14.17	9.17 ↓5.00	72.00	66.95 ↓5.05	36.64	31.35 ↓5.29
	Random-Dynamic*	26.59	20.23 ↓6.36	17.97	12.94 ↓5.03	75.85	69.92 ↓5.93	40.14	34.36 ↓5.78
	PPL-Top	25.04	17.92 ↓7.12	15.50	9.58 ↓5.92	76.65	71.55 ↓5.10	39.06	33.02 ↓6.04
	PPL-Middle	25.88	19.17 ↓6.71	16.25	11.25 ↓5.00	76.88	71.78 ↓5.10	39.67	34.07 ↓5.60
	CircuitSeer	25.92	19.50 ↓6.42	17.97	11.42 ↓6.55	73.00	66.30 ↓6.70	38.96	32.41 ↓6.55
	LIMR	28.34	22.08 ↓6.26	16.67	10.42 ↓6.25	78.03	72.45 ↓5.58	41.01	34.98 ↓6.03
	LearnAlign	29.17	22.91 ↓6.26	17.92	11.25 ↓6.67	78.90	72.38 ↓6.52	42.00	35.51 ↓6.49
	DEPO*	29.58	22.08 ↓7.50	18.34	12.92 ↓5.42	80.50	73.03 ↓7.47	42.81	36.01 ↓6.80
	MechDS-Static	32.08	25.00 ↓7.08	20.83	14.17 ↓6.66	83.00	77.80 ↓5.20	45.30	38.99 ↓6.31
	MechDS-Dynamic*	**34.17**	**28.75** ↓5.42	**22.50**	**18.33** ↓4.17	**84.48**	**79.90** ↓4.58	**47.05**	**42.33** ↓4.72
Qwen2.5-Math-1.5B-Instruct	Random	12.50	5.84 ↓6.66	12.08	5.83 ↓6.25	74.90	68.88 ↓6.03	33.16	26.84 ↓6.31
	Random-Dynamic*	15.85	8.60 ↓7.25	15.02	8.31 ↓6.71	77.73	71.34 ↓6.39	36.20	29.42 ↓6.78
	PPL-Top	13.75	5.42 ↓8.33	12.50	6.25 ↓6.25	75.28	69.20 ↓6.08	33.84	26.96 ↓6.88
	PPL-Middle	14.17	6.25 ↓7.92	12.92	7.09 ↓5.83	75.75	69.53 ↓6.23	34.28	27.62 ↓6.66
	CircuitSeer	14.40	7.74 ↓6.66	14.17	7.62 ↓6.55	75.88	69.43 ↓6.45	34.82	28.26 ↓6.56
	LIMR	15.42	8.75 ↓6.67	14.58	7.50 ↓7.08	76.35	70.00 ↓6.35	35.45	28.75 ↓6.70
	LearnAlign	17.09	11.25 ↓5.84	16.25	9.58 ↓6.67	77.90	70.95 ↓6.95	37.08	30.59 ↓6.49
	DEPO*	17.50	12.08 ↓5.42	16.67	10.42 ↓6.25	78.00	71.33 ↓6.68	37.39	31.27 ↓6.12
	MechDS-Static	20.83	15.42 ↓5.42	20.42	14.58 ↓5.83	81.33	76.30 ↓5.03	40.86	35.43 ↓5.43
	MechDS-Dynamic*	**22.08**	**18.75** ↓3.33	**21.25**	**17.08** ↓4.17	**81.98**	**77.95** ↓4.03	**41.77**	**37.93** ↓3.84
DeepSeek-R1-Distill-Llama-8B	Random	48.75	42.08 ↓6.67	37.50	31.66 ↓5.83	89.48	83.30 ↓6.18	58.58	52.35 ↓6.23
	Random-Dynamic*	50.67	44.86 ↓5.81	39.04	33.11 ↓5.93	91.30	85.23 ↓6.07	60.34	54.40 ↓5.94
	PPL-Top	48.34	41.67 ↓6.67	37.92	30.83 ↓7.08	89.85	83.60 ↓6.25	58.70	52.03 ↓6.67
	PPL-Middle	48.75	42.50 ↓6.25	38.75	32.50 ↓6.25	90.43	83.85 ↓6.58	59.31	52.95 ↓6.36
	CircuitSeer	49.96	43.32 ↓6.64	38.58	32.50 ↓6.08	89.11	82.65 ↓6.46	59.22	52.82 ↓6.39
	LIMR	51.67	45.00 ↓6.67	40.84	34.63 ↓6.21	91.53	84.85 ↓6.68	61.34	54.83 ↓6.52
	LearnAlign	53.75	47.50 ↓6.25	41.67	35.42 ↓6.25	91.65	85.70 ↓5.95	62.36	56.21 ↓6.15
	DEPO*	53.75	47.92 ↓5.83	42.08	35.84 ↓6.25	91.95	85.83 ↓6.13	62.59	56.53 ↓6.07
	MechDS-Static	56.67	50.00 ↓6.67	44.58	38.75 ↓5.83	93.13	86.08 ↓7.05	64.79	58.28 ↓6.52
	MechDS-Dynamic*	**57.09**	**53.33** ↓3.75	**45.42**	**41.25** ↓4.17	**94.05**	**88.10** ↓5.95	**65.52**	**60.89** ↓4.62

47.05 is obtained with 40 heads. This trend suggests that using too few heads may provide insufficient coverage of components important for reasoning, whereas including too many may introduce less informative heads and dilute the RCD signal. Overall, MechDS is relatively stable around the default setting, with $K_{\text{head}} = 40$ providing the strongest performance.

7.2 Data Composition Analysis

Signal composition. We first compare the two individual signals with their joint use. **BE-only** statically selects middle-BE samples, whereas **RCD-only** statically selects low-RCD samples. As shown at the top of Table 2, both variants underperform MechDS-Static and MechDS-Dynamic. RCD-only performs better than BE-only, but still falls clearly short of jointly considering both dimensions. These results indicate that BE and RCD capture complementary properties of RLVR samples. Behavioral entropy characterizes the model's observable uncertainty, while RCD provides additional information about the internal reasoning mechanisms supporting generation.

Stability. We next examine the role of the initial stability stage. As shown at the middle of Table 2, **w/o Stability** skips the initial Low-BE/High-RCD stage by setting $\rho(\tau) = 1$ from the beginning. Although its Mean@16 slightly increases from 47.05 to 47.32, the Judge score drops from 42.33 to 40.66, enlarging the answer–reasoning gap from 4.75 to 6.66. This indicates that the initial stability stage primarily contributes to reasoning reliability rather than final answer accuracy. We further fix $\beta = 0.75$ and vary $\alpha \in \{0.125, 0.25, 0.375, 0.5\}$ to control the transition duration. As shown in Figure 5(b), increasing α beyond 0.25 progressively decreases both Mean@16 and Judge, suggesting that overly long transitions reduce the time spent in the intermediate Middle-BE/Low-RCD region and weaken subsequent optimization. A smaller $\alpha = 0.125$ yields slightly higher Mean@16 but a larger answer–reasoning gap, whereas $\alpha = 0.25$ achieves the highest Judge score with near-best Mean@16. These results suggest that a moderate transition duration provides a better balance between reliable early adaptation and subsequent learning.

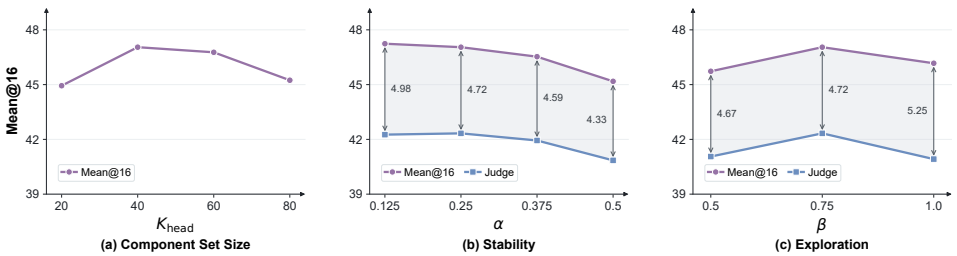

Figure 5: Average parameter sensitivity of MechDS on Qwen2.5-Math-7B.

Table 2: Average ablation results on Qwen2.5-Math-7B.

Variant	Mean@16	Judge
BE-only	40.28	34.95 ↓5.33
RCD-only	42.86	36.07 ↓6.79
MechDS-Static	45.30	38.99 ↓6.31
MechDS-Dynamic	47.05	**42.33** ↓4.75
- w/o Exploration	45.73	41.06 ↓**4.67**
- w/o Stability	**47.32**	40.66 ↓6.66
- w/ Random Heads	41.62	36.35 ↓5.27
- w/ Low Heads	40.16	34.37 ↓5.79

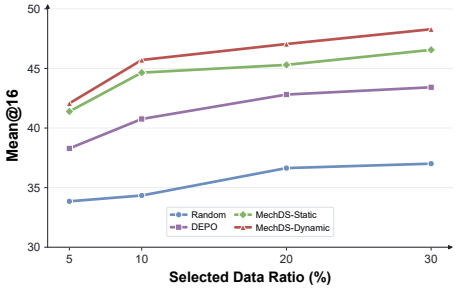

Figure 6: Average performance on Qwen2.5-Math-7B under different selected data ratios.

Exploration. We examine exploration in the late stage. As shown at the middle of Table 2, **w/o Exploration** sets $\beta = 0.5$, removing the transition toward samples with higher BE and low RCD. This reduces performance while leaving the answer–reasoning gap nearly unchanged, suggesting that late stage exploration provides learning opportunities. We fix $\alpha = 0.25$ and vary $\beta \in \{0.5, 0.75, 1.0\}$ to control exploration strength. As shown in Figure 5(c), increasing β from 0.5 to 0.75 improves both Mean@16 and Judge, whereas increasing it to 1.0 lowers the Judge score and enlarges the answer–reasoning gap. This indicates that moving beyond samples with middle BE can provide learning opportunities. However, pushing the target too far toward the region with the highest BE emphasizes uncertain samples, making adaptation less reliable and limiting reasoning quality.

7.3 Data Efficiency Analysis

The data efficiency analysis evaluates whether MechDS maintains its advantage under different selection budgets. We vary the per-epoch selected data ratio over $\{5\%, 10\%, 20\%, 30\%\}$ under otherwise identical training settings. As shown in Figure 6, MechDS-Dynamic consistently achieves the best performance across all budgets, while the fixed-subset MechDS-Static also consistently outperforms Random and DEPO. Notably, MechDS-Dynamic with a 10% per-epoch selection budget surpasses DEPO with a 30% budget. Moreover, the performance of MechDS-Static shows that the advantage of BE–RCD selection persists without stage-adaptive re-selection. These results demonstrate that MechDS remains effective across selection ratios and identifies more valuable RLVR training data under limited budgets.

8 Conclusion

We study RLVR data selection from a mechanistic perspective and introduce reasoning component dependence (RCD), which measures a sample's reliance on components critical to reasoning. Analysis with behavioral entropy (BE) shows that RCD distinguishes training value among samples with similar behavioral signals, revealing information missed by extrinsic proxies. These findings motivate MechDS, which adaptively selects RLVR data in the BE–RCD space. Across three models and three benchmarks, MechDS consistently improves answer accuracy and reasoning validity over baselines, demonstrating the value of internal component information for RLVR data selection.

Reproducibility Statement

We provide detailed model, training, data selection, component identification, and evaluation configurations in Section 4.1, Section 6.1 and Appendix A.

AI Use Statement

In this work, we used generative AI tools to provide feedback on research methodology and experimental design. We also used them to summarize and analyze existing literature, support information search and retrieval, and improve the readability of the manuscript. All AI-assisted work was reviewed and verified by the authors. We take responsibility for the final content of this work, including any text or claims produced with the aid of generative AI.

References

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, et al. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 6:16318–16352, 2025.

Zachary Ankner, Cody Blakeney, Kartik Sreenivasan, Max M Marion, Matthew Leavitt, and Mansheej Paul. Perplexed by perplexity: Perplexity-based data pruning with small reference models. In *International Conference on Learning Representations*, volume 2025, pp. 100615–100634, 2025.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.

GLM-5 Team, Aohan Zeng, Xin Lv, Zhenyu Hou, et al. GLM-5: From vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

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

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

Guan Zhe Hong, Nishanth Dikkala, Enming Luo, Cyrus Rashtchian, Xin Wang, and Rina Panigrahy. A implies b: Circuit analysis in llms for propositional logical reasoning. *Advances in Neural Information Processing Systems*, 38:30749–30803, 2026.

Geonhee Kim, Marco Valentino, and Andre Freitas. Reasoning circuits in language models: A mechanistic interpretation of syllogistic inference. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10074–10095, 2025.

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

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, et al. The bigscience roots corpus: A 1.6 tb composite multilingual dataset. *Advances in Neural Information Processing Systems*, 35:31809–31826, 2022.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7602–7635, 2024.

Shikun Li, Shipeng Li, Zhiqin Yang, Xinghua Zhang, Gaode Chen, Xiaobo Xia, Hengyu Liu, and Zhe Peng. Learnalign: Reasoning data selection for reinforcement learning in large language models based on improved gradient alignment. *arXiv e-prints*, pp. arXiv–2506, 2025a.

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

David Lindner, János Kramár, Sebastian Farquhar, Matthew Rahtz, Tom McGrath, and Vladimir Mikulik. Tracr: Compiled transformers as a laboratory for interpretability. *Advances in Neural Information Processing Systems*, 36:37876–37899, 2023.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition. `https://openai.com/index/gpt-5-6/`, 2026.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork. `https://qwen.ai/blog?id=qwen3.8`, August 2026.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Xinyu Tang, Zhenduo Zhang, Yurou Liu, Wayne Xin Zhao, Zujie Wen, Zhiqiang Zhang, and Jun Zhou. Towards high data efficiency in reinforcement learning with verifiable reward. *arXiv preprint arXiv:2509.01321*, 2025.

Jesse Vig and Yonatan Belinkov. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL workshop BlackboxNLP: analyzing and interpreting neural networks for NLP*, pp. 63–76, 2019.

Shaobo Wang, Yongliang Miao, Yuancheng Liu, Qianli Ma, Ning Liao, and Linfeng Zhang. Circuitseer: Mining high-quality data by probing mathematical reasoning circuits in llms. *arXiv preprint arXiv:2510.18470*, 2025.

Tingyu Xia, Bowen Yu, Kai Dang, An Yang, Yuan Wu, Yuan Tian, Yi Chang, and Junyang Lin. Rethinking data selection at scale: Random selection is almost all you need. *arXiv preprint arXiv:2410.09335*, 1, 2024.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

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

Ningyuan Yang, Weihua Du, Weiwei Sun, Sean Welleck, and Yiming Yang. Gradalign: Gradient-aligned data selection for llm reinforcement learning. *arXiv preprint arXiv:2602.21492*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

Zhenhong Zhou, Haiyang Yu, Xinghua Zhang, Rongwu Xu, Fei Huang, Kun Wang, Yang Liu, Junfeng Fang, and Yongbin Li. On the role of attention heads in large language model safety. In *International Conference on Learning Representations*, volume 2025, pp. 84042–84071, 2025.

A Detailed Experimental Setup

Models. We evaluate MechDS on three open-weight mathematical reasoning models covering different parameter scales and post-training histories. Qwen2.5-Math-7B (Yang et al., 2024) is a math-specialized base model from the Qwen2.5-Math family, whereas Qwen2.5-Math-1.5B-Instruct is a smaller instruction-tuned variant designed to follow mathematical instructions and generate step-by-step solutions. DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025) is an 8B reasoning model obtained by distilling reasoning capabilities from DeepSeek-R1 into Llama-3.1-8B. These models allow us to evaluate MechDS across different model scales, model families, and prior post-training recipes.

Component Identification and Signal Computation. For each backbone model, the model to be trained serves as both the reference model $\mathcal{M}_{\mathrm{ref}}$ and the trajectory generator. We first generate solution trajectories with this model and retain those that pass both the answer correctness and reasoning validity filters. The retained trajectories are ranked by path entropy, and the top $K_{\mathrm{prob}} = 500$ trajectories are used as the probe set. We then evaluate all attention heads using the causal ablation described in the main text and select the top $K_{\mathrm{head}} = 40$ heads according to their importance scores as the reference component set $\mathcal{H}^{\star}$. Component identification is performed once before RLVR training. For each candidate training sample, the same model first generates one solution trajectory autoregressively. We then evaluate this fixed trajectory under teacher forcing to compute both behavioral entropy (BE) and reasoning component dependence (RCD). RCD is obtained by comparing the next token distributions of the original and ablated models along the same trajectory, where all heads in $\mathcal{H}^{\star}$ are replaced by undifferentiated attention in the ablated model. Both BE and RCD are computed once before training and reused throughout the subsequent data selection process.

Training. For both Qwen2.5-Math-7B and Qwen2.5-Math-1.5B-Instruct, we implement GRPO training with `verl` (Sheng et al., 2025) and use `vLLM` (Kwon et al., 2023) for rollout generation. Each experiment is conducted on $8\times$ NVIDIA A100 GPUs with a global batch size of 256. The maximum prompt and response lengths are both set to 2,048 tokens, and eight responses are sampled for each prompt. The actor is optimized with a learning rate of 1×10^{-6}, a PPO mini-batch size of 256, and a per-GPU micro-batch size of 8. We use the low-variance KL loss with a coefficient of 0.001, set the entropy coefficient to zero, and do not include a KL penalty in the reward. Gradient checkpointing and padding removal are enabled. Rollout inference uses a tensor-parallel size of 2 and a GPU memory utilization ratio of 0.6. Each run is trained for at most 1,000 optimization steps, with checkpoints saved every 50 steps.

Data Selection. We use DAPO-Math (Yu et al., 2026) as the candidate training pool. All methods use the same 20% selection budget. For static methods, the selected subset is fixed before optimization and used throughout training. MechDS-Static selects the 20% samples closest to the fixed anchor $(0.5, 0)$. MechDS-Dynamic re-ranks the candidate pool at the beginning of each epoch along the stage-dependent anchor trajectory, with $\alpha = 0.25$ and $\beta = 0.75$ by default, while retaining the same 20% per-epoch selection budget.

Evaluation. We evaluate model performance on MATH-500 (Hendrycks et al., 2021), AIME 2024, and AIME 2025 using a separate evaluation pipeline. For each problem, we sample 16 responses and compute Mean@16 as the average final-answer correctness over these responses. We also evaluate reasoning validity on the same responses using DeepSeek-V4-Flash (Xu et al., 2026) as an LLM-as-Judge. For each answer-correct response, the Judge determines whether the reasoning is logically valid and genuinely supports the final conclusion; answer-incorrect responses are directly treated as reasoning-invalid. Judge is then computed as the percentage of all evaluated responses that are both answer-correct and reasoning-valid. The complete evaluation prompt is shown in Figure 7. We define the answer–reasoning gap as $\Delta = \mathrm{Mean@16} - \mathrm{Judge}$, where lower values indicate better answer–reasoning consistency. Evaluation generation uses a temperature of 0.7, top-p of 0.95, and a maximum response length of 2,048 tokens.

Baselines. We compare MechDS with seven representative data selection baselines. **Random** uniformly selects a fixed subset, while **Random-Dynamic** uniformly re-samples the subset at each epoch. **PPL-Top** (Laurençon et al., 2022) selects samples with the highest perplexity, whereas **PPL-Middle** (Ankner et al., 2025) selects samples around the middle of the perplexity distribu-

LLM-as-Judge

```
You are given a competition math problem, the KNOWN-CORRECT final answer, and a candidate solution that already reached this correct answer.

Judge ONLY the solution's REASONING PATH -- not the final answer (it is already known to be correct).

- VALID: the reasoning is logically sound and the answer genuinely FOLLOWS from the work. Abbreviating routine/obvious steps is fine.
- INVALID: the solution reaches the correct answer DESPITE flawed reasoning, e.g.:
  * a false claim or wrong assumption (asserting an identity/fact that doesn't hold)
  * an arithmetic error on a step that matters
  * an unjustified leap on a CRITICAL step
  * a self-contradiction
  * a fabricated / nonexistent formula
  * the final answer does NOT follow from the derivation (lucky guess / back-filled)
  * plugging in wrong values

Be strict: if a CRITICAL step is wrong, unjustified, or the answer doesn't follow, it is INVALID -- even though the final answer is correct. Verify the
key steps yourself before deciding. The solution may be in English or Chinese.

Output EXACTLY this, nothing after:
VERDICT: VALID|INVALID
ERROR_TYPE: none|false_claim|arithmetic|unjustified_leap|contradiction|fabricated_formula|answer_not_derived|wrong_substitution|other
REASON: <one short sentence; if INVALID, quote the specific bad step>

[PROBLEM]
{problem}

[KNOWN CORRECT ANSWER]
{answer}

[CANDIDATE SOLUTION]
{solution}
```

Figure 7: Prompt used for LLM-as-Judge reasoning-validity evaluation.

tion. **CircuitSeer** (Wang et al., 2025) identifies reasoning circuits through attention head ablation and scores samples by the attention variance over these circuits, followed by score-proportional soft sampling. **LIMR** (Li et al., 2025b) measures learning impact by the alignment between each sample's reward trajectory and the model's overall learning trajectory. **LearnAlign** (Li et al., 2025a) combines success-rate-based learnability with projected gradient alignment to select learnable and representative samples. **DEPO** (Tang et al., 2025) combines offline selection based on diversity, influence, and difficulty with online explorability-guided filtering and replay. All methods use the same candidate pool and selection budget.

B Runtime and Computational Cost

We further evaluate the computational cost of data selection under the same hardware environment. All methods are measured on the same $8\times$ NVIDIA A100-SXM4-80GB node using the same candidate pool. We report both wall-clock time and GPU hours for the complete data selection process. The downstream RLVR training cost is excluded, since the same training configuration is used after selection. For MechDS, the reported cost includes reasoning component identification, BE and RCD computation, and subsequent data selection. As shown in Table 3, MechDS requires 6h49m of wall-clock time, corresponding to 54.5 GPU-h. Its computational cost is close to DEPO, while remaining substantially lower than LearnAlign and LIMR. In particular, MechDS requires only 12 additional minutes compared with DEPO, despite incorporating model internal information through component identification and dependence estimation. These results indicate that the additional mechanistic analysis in MechDS introduces moderate computational overhead while maintaining competitive data selection efficiency.

C Cross-Judge Consistency Analysis

The appendix evaluate the robustness of our reasoning validity assessment using three additional LLM judges from different model families: GPT-5.6 Luna (OpenAI, 2026), GLM-5.3 (GLM-5 Team et al., 2026), and Qwen3.8-Max (Qwen Team, 2026). We sample 100 responses from the evaluation results, comprising 70 responses labeled valid and 30 labeled invalid by the primary

Table 3: Computational cost of data selection on the same $8\times$ A100-SXM4-80GB node. GPU-h denotes the total GPU hours consumed during data selection.

Method	Wall-clock	GPU-h
Random	< 1s	< 0.01
PPL-Top	4h46m	38.1
PPL-Middle	4h46m	38.1
LIMR	48h22m	386.9
LearnAlign	9h13m	73.7
DEPO	6h37m	52.9
MechDS	**6h49m**	**54.5**

judge, DeepSeek V4-Flash. All alternative judges evaluate the same responses using the same validity criterion and evaluation prompt. The agreement rates with DeepSeek V4-Flash are 92%, 98%, and 98%, respectively. The consistently high agreement across different model families suggests that the reasoning validity evaluation is robust to the choice of LLM judge.

D LIMITATIONS

Firstly, our method identifies components that are important for valid mathematical reasoning, but we do not claim that these components are used only for reasoning. They may also contribute to other language or mathematical generation tasks. Future work will further examine their functional specificity through controlled comparisons and targeted causal interventions. **Then**, our current implementation focuses on attention heads as the unit of mechanistic analysis. Reasoning may also rely on other internal structures, such as MLP features or more distributed computations, which are not captured by the present RCD formulation. Extending RCD to richer component types, and examining whether multiple mechanistic signals are complementary, is a natural next step. **Finally**, both BE and RCD are computed once before RLVR training and remain unchanged throughout optimization. As the policy and its internal representations evolve, these pre-computed signals may not fully reflect the model state at later training stages. Recomputing BE and RCD during training could provide more up-to-date signals, but would substantially increase the computational cost of data selection. We therefore compute these signals only once before training, while retaining dynamic selection through the stage-dependent moving anchor. Future work will explore lightweight signal updates that better track model changes without incurring excessive overhead.

E CASE STUDY

Table 4 presents a representative comparison. MechDS-Dynamic provides both the correct answer and valid reasoning, whereas MechDS-Static only reaches the correct answer and DEPO fails on both. This case highlights that final-answer correctness alone can conceal substantial differences in reasoning quality. In particular, a model may obtain the correct outcome through incomplete, unsupported, or internally inconsistent reasoning, which is indistinguishable under answer-only evaluation. MechDS-Dynamic succeeds at both the answer and reasoning levels, whereas the static variant improves answer correctness without fully closing the answer–reasoning gap. This suggests that the stage-adaptive selection strategy does not merely increase the probability of reaching the correct answer, but also encourages more reliable and self-consistent reasoning processes. This observation is consistent with the quantitative results in Table 1, where MechDS-Dynamic achieves a higher Judge score and a smaller answer–reasoning gap. The following examples illustrate these differences in detail.

E.1 ROOTS-OF-UNITY CASE

Problem. If $\omega^{1997} = 1$ and $\omega \neq 1$, then evaluate $\dfrac{1}{1+\omega} + \dfrac{1}{1+\omega^2} + \cdots + \dfrac{1}{1+\omega^{1997}}$.

MechDS-Dynamic. **Answer:** 998.5; **Correct:** Yes; **Judge:** Yes.

To solve the given problem, we need to evaluate the sum $\sum_{k=1}^{1997} \frac{1}{1+\omega^k}$, where $\omega^{1997} = 1$ and $\omega \neq 1$. This means ω is a primitive 1997th root of unity.

First, let's consider the properties of the roots of unity. The 1997th roots of unity are the solutions to the equation $x^{1997} = 1$. These solutions are $1, \omega, \omega^2, \ldots, \omega^{1996}$, where $\omega = e^{2\pi i/1997}$.

We need to evaluate the sum $\sum_{k=1}^{1997} \frac{1}{1+\omega^k}$. Notice that $\omega^{1997} = 1$, so $\omega^{1997+k} = \omega^k$. This means the terms ω^k for $k = 0, 1, 2, \ldots, 1996$ are all distinct and form a complete set of 1997th roots of unity.

We can pair the terms in the sum as follows: $\sum_{k=1}^{1996} \frac{1}{1+\omega^k} + \frac{1}{1+\omega^{1997}}$. Since $\omega^{1997} = 1$, the last term simplifies to $\frac{1}{1+1} = \frac{1}{2}$.

Next, we need to evaluate the sum $\sum_{k=1}^{1996} \frac{1}{1+\omega^k}$. To do this, we use the fact that for any k, $\frac{1}{1+\omega^k} + \frac{1}{1+\omega^{1997-k}} = \frac{1}{1+\omega^k} + \frac{1}{1+\omega^{-k}} = \frac{1}{1+\omega^k} + \frac{\omega^k}{\omega^k+1} = \frac{1+\omega^k}{1+\omega^k} = 1$.

This means we can pair the terms as follows: $\left(\frac{1}{1+\omega} + \frac{1}{1+\omega^{1996}}\right) + \left(\frac{1}{1+\omega^2} + \frac{1}{1+\omega^{1995}}\right) + \cdots + \left(\frac{1}{1+\omega^{998}} + \frac{1}{1+\omega^{999}}\right)$. There are 998 such pairs, each summing to 1, so the sum of these pairs is $998 \cdot 1 = 998$. Adding the last term $\frac{1}{2}$, we get $998 + \frac{1}{2} = \frac{1996}{2} + \frac{1}{2} = \frac{1997}{2}$. Therefore, the final answer is $\boxed{\frac{1997}{2}}$.

MechDS-Static. **Answer:** 998.5; **Correct:** Yes; **Judge:** No.

To solve the problem, we need to evaluate the sum $\sum_{k=1}^{1997} \frac{1}{1+\omega^k}$, where ω is a 1997th root of unity, i.e., $\omega^{1997} = 1$ and $\omega \neq 1$.

First, let's consider the properties of the roots of unity. The 1997th roots of unity are the solutions to the equation $x^{1997} = 1$. These solutions are $1, \omega, \omega^2, \ldots, \omega^{1996}$, where $\omega = e^{2\pi i/1997}$.

We can use the fact that the sum of the reciprocals of $1+\omega^k$ for all 1997th roots of unity (excluding 1) can be simplified using the symmetry of the roots of unity. Specifically, we can use the fact that the sum of the reciprocals of $1+\omega^k$ for all k from 1 to 1996 is equal to $\frac{1997-1}{2} = 998$.

Here's the step-by-step reasoning:

1. Consider the polynomial $P(x) = x^{1997} - 1$. The roots of this polynomial are $1, \omega, \omega^2, \ldots, \omega^{1996}$.

2. We want to evaluate the sum $\sum_{k=1}^{1997} \frac{1}{1+\omega^k}$. Note that $\omega^{1997} = 1$, so $\omega^{1997} - 1 = 0$.

3. Using the properties of roots of unity, we can use the fact that the sum of the reciprocals of $1+\omega^k$ for all k from 1 to 1996 is equal to $\frac{1997-1}{2} = 998$.

Therefore, the sum is $\sum_{k=1}^{1997} \frac{1}{1+\omega^k} = 998.5$.

However, since we are summing over all 1997 roots of unity (including 1), we need to adjust for the fact that $\omega^0 = 1$ is not included in the sum. The correct sum is actually 998.5, but since we are summing over 1997 terms, the correct answer is 998.5.

Thus, the final answer is $\boxed{998.5}$.

DEPO. **Answer:** 998; **Correct:** No; **Judge:** No.

Given that $\omega^{1997} = 1$ and $\omega \neq 1$, ω is a primitive 1997th root of unity. We need to evaluate the sum $S = \frac{1}{1+\omega} + \frac{1}{1+\omega^2} + \cdots + \frac{1}{1+\omega^{1997}}$.

First, observe that $\omega^{1997} = 1$ implies $\omega^{1997k} = 1$ for any integer k. Therefore, $\omega^{1997} = \omega^0 = 1$. This means that ω^{1997} is the same as ω^0, and the terms in the sum repeat every 1997 terms. Hence, we can rewrite the sum as $S = \sum_{k=1}^{1997} \frac{1}{1+\omega^k}$.

To evaluate this sum, we use the fact that the sum of the reciprocals of $1 + \omega^k$ for all 1997th roots of unity (including 1) can be simplified using properties of roots of unity. Specifically, we use the identity for the sum of reciprocals of $1 + \omega^k$: $\sum_{k=0}^{1996} \frac{1}{1+\omega^k} = \frac{1997}{2}$.

This identity can be derived from the fact that the polynomial $x^{1997} - 1$ has roots ω^k for $k = 0, 1, 2, \ldots, 1996$, and by considering the polynomial $(x-1)^{1997} - 1$, which has roots $1 + \omega^k$ for $k = 0, 1, 2, \ldots, 1996$.

Since $\omega^0 = 1$, we need to exclude the term $\frac{1}{1+1} = \frac{1}{2}$ from the sum and add it separately. Therefore, the sum we are interested in is $S = \left(\sum_{k=0}^{1996} \frac{1}{1+\omega^k}\right) - \frac{1}{2} = \frac{1997}{2} - \frac{1}{2} = \frac{1996}{2} = 998$.

Thus, the value of the sum is $\boxed{998}$.

E.2 Quadratic-Maximization Case

Problem. What is the maximum value of $4(x+7)(2-x)$, over all real numbers x?

MechDS-Dynamic. **Answer:** 81; **Correct:** Yes; **Judge:** Yes.

To find the maximum value of the expression $4(x+7)(2-x)$, we can follow these steps:

1. Expand the expression.

2. Rewrite the expression in the form of a quadratic equation.

3. Find the vertex of the quadratic equation, which will give us the maximum value since the parabola opens downwards.

Let's start by expanding the expression:

$4(x+7)(2-x) = 4(x\cdot 2 + x\cdot(-x) + 7\cdot 2 + 7\cdot(-x)) = 4(2x - x^2 + 14 - 7x) = 4(-x^2 - 5x + 14) = -4x^2 - 20x + 56$.

The expression is a quadratic function of the form $ax^2 + bx + c$, where $a = -4$, $b = -20$, and $c = 56$. Since the coefficient of x^2 is negative ($a = -4$), the parabola opens downwards, and the maximum value occurs at the vertex.

The x-coordinate of the vertex of a parabola given by $ax^2 + bx + c$ is found using the formula $x = -\frac{b}{2a}$:

$x = -\frac{-20}{2\cdot(-4)} = -\frac{20}{8} = -\frac{5}{2}$.

Now, we substitute $x = -\frac{5}{2}$ back into the original expression to find the maximum value:

$$4\left(-\frac{5}{2}+7\right)\left(2-\left(-\frac{5}{2}\right)\right) = 4\left(-\frac{5}{2}+\frac{14}{2}\right)\left(2+\frac{5}{2}\right) = 4\left(\frac{9}{2}\right)\left(\frac{9}{2}\right) = 4\cdot\frac{81}{4} = 81.$$

So, the maximum value of the expression $4(x+7)(2-x)$ is $\boxed{81}$.

MechDS-Static. **Answer:** 81; **Correct:** Yes; **Judge:** No.

To find the maximum value of the expression $4(x + 7)(2 - x)$, we can start by expanding and rewriting it in a standard quadratic form. The expression can be expanded as follows:

$$4(x+7)(2-x) = 4(x^2+7x-2x-14) = 4(x^2+5x-14) = 4x^2+20x-56.$$

This is a quadratic expression of the form $ax^2 + bx + c$, where $a = 4$, $b = 20$, and $c = -56$. The maximum or minimum value of a quadratic expression $ax^2 + bx + c$ occurs at $x = -\frac{b}{2a}$. Since the coefficient of x^2 (which is $a = 4$) is positive, the quadratic expression opens upwards, meaning it has a minimum value if we were looking for that. However, since we are asked for the maximum value, we need to evaluate the expression at the vertex, which is the critical point for the maximum value in this context.

The vertex (critical point) occurs at:

$$x = -\frac{b}{2a} = -\frac{20}{2\cdot 4} = -\frac{20}{8} = -\frac{5}{2}.$$

Now, we substitute $x = -\frac{5}{2}$ back into the original expression to find the maximum value:

$$4\left(-\frac{5}{2}+7\right)\left(2-\left(-\frac{5}{2}\right)\right) = 4\left(\frac{-5+14}{2}\right)\left(\frac{4+5}{2}\right) = 4\left(\frac{9}{2}\right)\left(\frac{9}{2}\right) = 4\cdot\frac{81}{4} = 81.$$

So, the maximum value of the expression $4(x+7)(2-x)$ is $\boxed{81}$.

DEPO. **Answer:** 81; **Correct:** Yes; **Judge:** No.

To find the maximum value of the expression $4(x + 7)(2 - x)$, we can start by expanding and rewriting it in a standard quadratic form. The expression can be expanded as follows:

$$4(x+7)(2-x) = 4(x^2+7x-2x-14) = 4(x^2+5x-14) = 4x^2+20x-56.$$

This is a quadratic expression of the form $ax^2 + bx + c$, where $a = 4$, $b = 20$, and $c = -56$. The maximum or minimum value of a quadratic expression $ax^2 + bx + c$ occurs at $x = -\frac{b}{2a}$. Since the coefficient of x^2 (which is $a = 4$) is positive, the quadratic expression opens upwards, meaning it has a minimum value if we were looking for that. However, since we are asked for the maximum value, we need to recognize that the expression $4(x+7)(2-x)$ is a product of a constant 4 and a quadratic expression that opens downwards (because the leading coefficient of the quadratic part is negative). Therefore, the maximum value occurs at the vertex of the quadratic expression $x^2 + 5x - 14$.

The vertex form for the maximum value of $4(x + 7)(2 - x)$ occurs at $x = -\frac{20}{2\cdot 4} = -\frac{20}{8} = -\frac{5}{2}$.

Now, we substitute $x = -\frac{5}{2}$ back into the original expression to find the maximum value:

$$4\left(-\frac{5}{2}+7\right)\left(2-\left(-\frac{5}{2}\right)\right) = 4\left(\frac{-5+14}{2}\right)\left(\frac{4+5}{2}\right) = 4\left(\frac{9}{2}\right)\left(\frac{9}{2}\right) = 4\cdot\frac{81}{4} = 81.$$

So, the maximum value of $4(x + 7)(2 - x)$ is $\boxed{81}$.

E.3 Discussion

Across both examples, answer correctness alone does not reliably reflect reasoning validity: a model may reach the correct result despite unsupported or internally inconsistent derivations. MechDS-Dynamic instead produces correct and self-consistent reasoning in both cases, consistent with its higher Judge scores and smaller answer–reasoning gaps in Table 1. These examples provide quali-

Table 4: Qualitative comparison of reasoning quality. Judge indicates whether the reasoning is considered valid by an LLM-as-Judge. MechDS-Dynamic succeeds in both answer correctness and reasoning validity, whereas MechDS-Static only reaches the correct answer and DEPO fails on both.

Problem. If $\omega^{1997} = 1$ and $\omega \neq 1$, evaluate $\frac{1}{1+\omega} + \frac{1}{1+\omega^2} + \cdots + \frac{1}{1+\omega^{1997}}$.

Method	Answer	Correct	Judge	Reasoning Summary
DEPO	998	✗	✗	It correctly identifies the full roots-of-unity sum as $1997/2$, but incorrectly removes the $k = 0$ term. The target sum already contains the equivalent term $k = 1997$, so this subtraction leads to the wrong answer.
MechDS-Static	998.5	✓	✗	It states without justification that $\sum_{k=1}^{1996}(1 + \omega^k)^{-1} = 998$ and then gives a confused adjustment argument. The final answer is correct, but the derivation does not properly establish it.
MechDS-Dynamic	**998.5**	✓	✓	It pairs the terms indexed by k and $1997-k$, showing that each pair sums to 1. The 1996 nontrivial terms form 998 pairs, and the remaining $k = 1997$ term contributes $1/2$.

tative evidence that stage-adaptive data selection improves not only final-answer accuracy but also reasoning reliability.