Semantic-Space Exploration and Exploitation in RLVR for LLM Reasoning

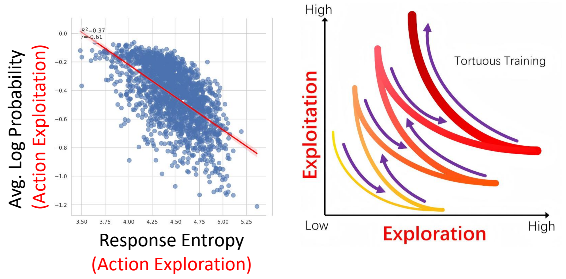
Fanding Huang[1*], Guanbo Huang[1*], Xiao Fan[1], Yi He[1], Xiao Liang[2], Xiao Chen[1]
Qinting Jiang[1], Faisal Nadeem Khan[1], Jingyan Jiang[3†], Zhi Wang[1†]
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]University of California, Los Angeles [3]Shenzhen Technology University
jiangjingyan@sztu.edu.cn, wangzhi@sz.tsinghua.edu.cn

Abstract

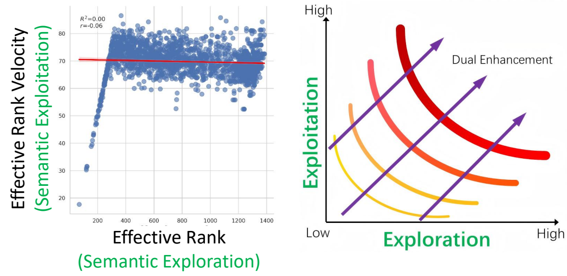
Reinforcement Learning with Verifiable Rewards (RLVR) for LLM reasoning is often framed as balancing exploration and exploitation in action space, typically operationalized with token-level proxies (e.g., output entropy or confidence). We argue that this apparent trade-off is largely a measurement artifact: token-level statistics reflect next-token uncertainty rather than how reasoning progresses over multi-token semantic structures. We therefore study exploration and exploitation in the hidden-state space of response trajectories. We use *Effective Rank* (ER) to quantify representational exploration and introduce its temporal derivatives, *Effective Rank Velocity* (ERV) and *Effective Rank Acceleration* (ERA), to characterize exploitative refinement dynamics. Empirically and theoretically, ER and ERV exhibit near-zero correlation in semantic space, suggesting the two capacities can be improved simultaneously. Motivated by this, we propose *Velocity-Exploiting Rank Learning* (VERL), which shapes the RLVR advantage with an auxiliary signal derived from ER/ERV and uses the more stable ERA as a meta-control variable to adaptively balance the incentives. Across multiple base models, RLVR algorithms, and reasoning benchmarks, VERL yields consistent improvements, including large gains on challenging tasks (e.g., 21.4% in Gaokao 2024). The code is available at `https://github.com/hf618/VERL`.

1 Introduction

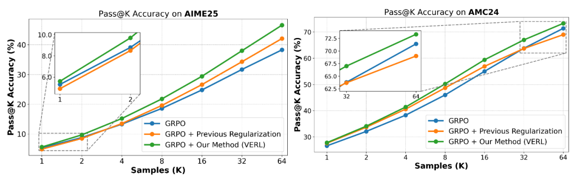
Recent advancements in Reinforcement Learning with Verifiable Rewards (RLVR) have substantially improved the reasoning abilities of Large Language Models (LLMs). A dominant narrative in recent work (Chen et al., 2025b; Yue et al., 2025; Deng et al., 2025a; Agarwal et al., 2025) interprets this progress through the lens of balancing *exploration* (seeking diverse reasoning paths) and *exploitation* (refining the most promising known strategies). In practice, however, these notions are almost exclusively operationalized in the *token-level action space*: exploration is associated with high-entropy next-token distributions, while exploitation



(a) Exploration & Exploitation in action space



(b) Exploration & Exploitation in semantic space



(c) Performance comparison with regularization

Figure 1: Comparative analysis of responses from DeepSeek-R1-Distill-Qwen-7B on simpleRL test set (Zeng et al., 2025). (a) Traditional metrics for exploitation & exploration constrained by negative coupling, leading to meandering progress for both capabilities. (b) Our hidden-state metrics are structurally decoupled, capturing complementary aspects of semantic exploration and exploitation. (c) Training regularization with our metrics demonstrates stronger performance in both exploitation (small K) and exploration (large K).

[*]Equal contribution: Fanding Huang and Guanbo Huang.
[†]Corresponding authors: Jingyan Jiang and Zhi Wang.

Findings of the Association for Computational Linguistics: ACL 2026, pages 38402–38449
July 2-7, 2026 ©2026 Association for Computational Linguistics

is linked to high-confidence ones. These token-level proxies are convenient to compute, yet they primarily measure uncertainty about the *next token*. As a result, the widely discussed "exploration–exploitation trade-off" (Fig. 1a) in reasoning may reflect the limitations of these proxies, instead of an inherent property of reasoning behavior.

This mismatch becomes more evident when distinguishing the *action space* from the *semantic space* in LLM reasoning. Reasoning progress is better viewed as transitions between *semantic states*, traversing concepts, subgoals, and partial derivations that span many tokens (Wei et al., 2022; Yao et al., 2023), while token sampling occurs in the action space. Token-level proxies (e.g., next-token entropy) can therefore misattribute local sampling uncertainty to semantic progress, leading to two failure modes: higher entropy may only amplify lexical variability without reaching new semantic directions, whereas lower entropy may collapse the trajectory too early into a familiar semantic region (Fu et al., 2025; Qiao et al., 2025; Agarwal et al., 2025).

Prior work has shown that pre-trained LLM hidden states naturally encode rich semantic structure (Huang et al., 2025; Jing et al., 2025) and reasoning dynamics (Cheng et al., 2026; Deng et al., 2025b). These findings suggest that hidden states can serve as a continuous representation of the model's evolving semantic state. Thus, this motivates that exploration and exploitation should be defined over trajectories in semantic space (how broadly the representation spans semantic directions, and how it refines within a region), rather than over token distributions in action space. We therefore ask: *Is the exploration–exploitation trade-off intrinsic to reasoning, or largely an artifact of measuring it in the token-level action space?*

To answer this, we step out of the action space and study exploration and exploitation directly in a *semantic space* induced by transformer hidden-state representations. In this semantic space, we quantify *representational exploration* by Effective Rank (ER), which measures how broadly a hidden-state trajectory spans semantic directions (Valeriani et al., 2023; Matthews et al., 2024; Jing et al., 2025). Following this intuition, we quantify *representational exploitation* by the temporal evolution of this span: Effective Rank Velocity (ERV) captures how semantic complexity is refined along the trajectory, and Effective Rank Acceleration (ERA) captures whether this refinement is sustaining, accelerating, or saturating. Equipped with these semantic-trajectory measures, we uncover a striking empirical result: ER and ERV exhibit near-zero correlation (Fig. 1b). This provides evidence that the commonly perceived exploration–exploitation trade-off is not intrinsic to LLM reasoning, but largely an artifact of measuring both notions in the token-level action space. Moreover, the two capacities can be *decoupled and improved simultaneously* in semantic space, as shown in Fig. 1c.

Building on this insight, we propose *Velocity-Exploiting Rank Learning* (VERL), a plug-and-play advantage-shaping method that optimizes exploration and exploitation *in semantic space*. VERL augments the RL advantage with auxiliary incentives derived from ER (encouraging broader semantic directions to avoid premature collapse) and ERV (reinforcing productive semantic refinement). Crucially, VERL uses ERA as a meta-control signal to adapt the strength of these incentives over training, as ERA tracks the trend of semantic refinement (i.e., whether exploitation is accelerating or saturating) and is theoretically more stable under short-term fluctuations (Sec. 3). Across diverse base models, RL algorithms, and reasoning benchmarks, VERL yields consistent improvements across evaluation settings, including up to 21.4% absolute accuracy gain on the challenging Gaokao 2024 benchmark.

Our contributions are summarized as follows:

- We recast exploration and exploitation from the token-level action space to a hidden-state semantic space, introducing ER, ERV, and ERA to measure semantic trajectories.
- We show that ER and ERV are nearly uncorrelated in the semantic space. This evidence suggests that the commonly reported trade-off is largely induced by token-level measurements rather than being an intrinsic constraint of reasoning.
- We present VERL, a plug-and-play method that improves both exploration and exploitation in the semantic space, and show consistent gains across models, RL algorithms, and benchmarks.

2 Preliminaries

2.1 Problem Formulation and Notations

We adopt an RLVR perspective for LLM reasoning. Given a prompt $x \sim \mathcal{P}_x$, an LLM policy $\pi_\theta(\cdot \mid x)$

generates a trajectory $y_{0:T} = (y_0, \dots, y_T)$ with $T \leqslant L_{\max}$. A scalar reward function $r(x, y)$ is used to evaluate the quality of the completed trajectory y given the prompt x. We aim to learn the parameters θ by maximizing the expected reward, where ϕ represents the optimal solution:

$$\phi = \underset{\theta}{\operatorname{argmax}} \, \mathbb{E}_{x \sim \mathcal{P}_x} \mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \left[r(x, y) \right]. \quad (1)$$

A common view (Chen et al., 2025b; Cheng et al., 2026; Deng et al., 2025a) is that optimizing this objective entails an exploration–exploitation trade-off: the policy must explore diverse paths to discover high-reward solutions while exploiting reliable strategies that consistently succeed.

2.2 Reinforcement Learning Baselines

Proximal Policy Optimization (PPO) (Schulman et al., 2017) is an RL algorithm that seeks to maximize a clipped surrogate objective, which prevents excessively large policy updates that may destabilize training. The objective is defined as:

$$\mathcal{L}_{\text{PPO}}(\theta) = \mathbb{E}_{x,y} \Big[\sum_{t=1}^{|y|} \min\big(\rho_t(\theta) A_t, \ \bar{\rho}_t(\theta) A_t\big) \Big],$$
$$\bar{\rho}_t(\theta) := \operatorname{clip}_{1-\epsilon_{\text{low}}}^{1+\epsilon_{\text{high}}}\big(\rho_t(\theta)\big), \quad (2)$$

where $\rho_t(\theta) := \frac{\pi_\theta(y_t|x, y_{<t})}{\pi_{\theta_{\text{old}}}(y_t|x, y_{<t})}$ is the probability ratio between the current and old policies, and A_t is the estimated advantage, often calculated using Generalized Advantage Estimation (GAE) (Schulman et al., 2016), with clipping thresholds ϵ.

Group Relative Policy Optimization (GRPO) from (Shao et al., 2024) computes a baseline directly from the rewards of multiple trajectories. For a given prompt, GRPO samples a group of G responses, obtains their rewards $\{r_i\}_{i=1}^G$, and normalizes them to compute the group-relative advantage for each response:

$$A_i := \big(r_i - \text{mean}(\{r_j\}_{j=1}^G)\big) / \text{std}(\{r_j\}_{j=1}^G). \quad (3)$$

This response-level advantage is then uniformly propagated to all tokens in the output sequence and optimized using the PPO-style objective in Eq. 2.

2.3 Hidden State Representations

Response hidden states. For an autoregressive response of length T, let $z_t \in \mathbb{R}^D$ denote the final hidden state at token t. Stacking them yields $\mathbf{Z} = [z_1; \dots; z_T] \in \mathbb{R}^{T \times D}$, which we view as a semantic trajectory. **Dataset hidden states.** Given N prompts, we summarize the i-th response by $\bar{z}_i = \frac{1}{T_i} \sum_{t=1}^{T_i} z_{i,t}$ (Skean et al., 2025), and stack them into $\bar{\mathbf{Z}} = [\bar{z}_1; \dots; \bar{z}_N] \in \mathbb{R}^{N \times D}$ to represent the dataset-level semantic distribution.

3 A Hidden-State Perspective on Representational Dynamics

Assumption. Building on prior studies (Skean et al., 2025; Jing et al., 2025), which show that linguistic and factual properties can be recovered from hidden states, our analysis assumes that semantic factors are approximately linearly decodable from hidden states. Our theoretical analysis is based on this assumption about representation.

3.1 Semantic Exploration: Effective Rank (ER)

We use Effective Rank (ER) as a proxy for semantic diversity, as it increases when hidden states are more evenly distributed across non-redundant latent directions and decreases when they collapse onto a few dominant ones. Following (Roy and Vetterli, 2007), we define the Effective Rank of a response hidden-state matrix $\mathbf{Z} \in \mathbb{R}^{T \times D}$ using its (non-padding) singular values $\{\sigma_j\}$. Let $p_j = \sigma_j / \sum_k \sigma_k$, then

$$\text{ER}(\mathbf{Z}) := \text{erank}(\mathbf{Z}) = \exp\Big[-\sum_j p_j \log p_j \Big]. \quad (4)$$

We interpret ER as *representational breadth* in the hidden-state semantic space: larger ER indicates that the trajectory spans a broader set of semantic directions, while smaller ER suggests concentration in fewer directions.

Theorem 3.1. *Suppose we have a matrix of embeddings $\mathbf{Z} \in \mathbb{R}^{T \times D}$. Then the ER of $\mathbf{Z}$ is a lower bound of conventional rank of $\mathbf{Z}$:*

$$1 \leqslant \text{erank}(\mathbf{Z}) \leqslant \text{rank}(\mathbf{Z}) \leqslant \min\{T, D\}. \quad (5)$$

Remark 3.2. While rank is discrete and blind to spectral concentration, ER is a continuous effective-dimensionality measure that grows as the singular spectrum flattens. Details are in Appendix. E.2.

3.2 Semantic Exploitation: Effective Rank Velocity (ERV) and Acceleration (ERA)

ER provides a *static* measure of how broadly a hidden-state trajectory spans semantic directions.

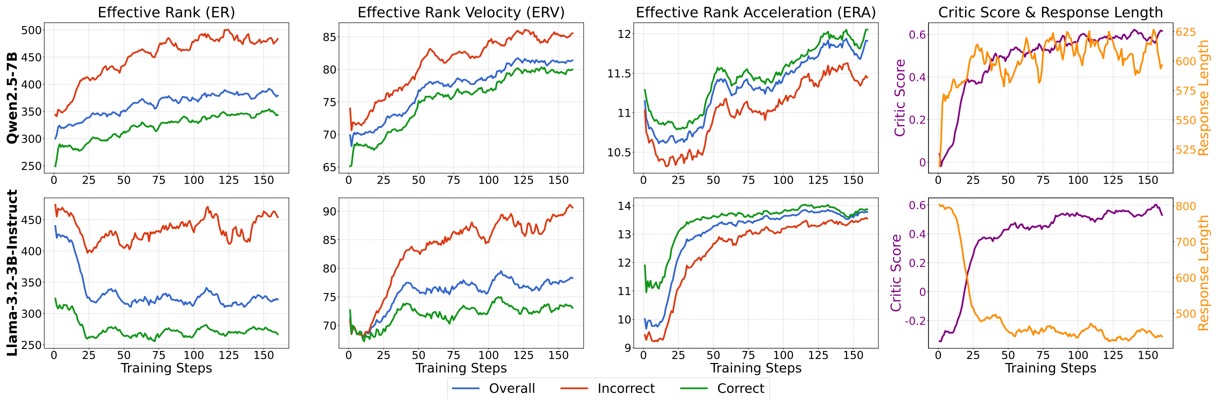


Figure 2: Response-level metrics during GRPO post-training, smoothed with a 10-step rolling window. Metrics are shown for the Overall batch, as well as for subsets of Correct and Incorrect samples. The rightmost column displays the average Critic Score (reward) and Response Length per batch.

To capture how this semantic breadth evolves during reasoning—whether the trajectory continues to expand into new directions, steadily refines within a region, or gradually saturates—we introduce two temporal-difference metrics: Effective Rank Velocity (ERV) and Effective Rank Acceleration (ERA). Importantly, these dynamics are defined *in semantic space* through hidden-state representations rather than in the token-level action space. This choice aligns with our view that exploitation reflects *productive semantic refinement*, which is more than just low-entropy sampling.

Setup. Let M be any trajectory metric computed on hidden states (like $M = \text{ER}$). Given a response of length T, let m_t denote the value of M computed on the prefix hidden-state matrix up to position t. To reduce sensitivity to short-range token fluctuations and control computation, we evaluate the metric every s tokens, where s is the temporal stride. Then, define the evaluation indices as $\mathcal{T} = \{s, 2s, \ldots, Ks\}$, where $K = \lfloor (T-1)/s \rfloor$.

Definition 3.3 (ERV, denoted by $\Delta_M^{(1)}$, as the *average historical-baseline deviation*)**.** For each step $j \in \mathcal{T}$, we define the per-step historical-baseline deviation from the historical-prefix average:

$$\delta_{j\cdot s} := m_{j\cdot s} - \frac{1}{j-1}\sum_{k=1}^{j-1} m_{k\cdot s}. \tag{6}$$

It measures whether current prefixes tend to exceed earlier-prefix averages, rather than the one-step change $m_t - m_{t-1}$. Thus, this form contrasts the current semantic breadth with the historical average over earlier prefixes. This tracks net semantic progress against running history, aligning with exploitation as greedy trajectory refinement under the current policy guidance (Appendix. E.3). A large positive $\delta_{j\cdot s}$ indicates that the trajectory at step $j \cdot s$ expands semantic breadth beyond its previous trend; a small value suggests that semantic expansion is no longer outpacing the past.

Then, we average these $\delta_{j\cdot s}$ of the response to obtain the *average historical-baseline deviation* for the whole reasoning trajectory:

$$\Delta_M^{(1)} := \frac{1}{K-1}\sum_{j=2}^{K} \delta_{j\cdot s}. \tag{7}$$

We specialize it as $\text{ERV} := \Delta_{\text{ER}}^{(1)}$, and interpret it as a measure of *exploitation in semantic space* during reasoning: sustained positive ERV reflects consistent representational refinement along the current reasoning path, whereas small ERV indicates diminishing returns and potential stagnation.

To quantify the changes in semantic breadth across different steps, we define the consecutive-step increments as $\Delta m_{r\cdot s} := m_{r\cdot s} - m_{(r-1)\cdot s}$, where $r \in \mathcal{T}$. This represents the difference in the metric M at every s-th step of the trajectory. We can then rearrange $\delta_{j\cdot s}$, the historical-baseline deviation, as follows:

$$\delta_{j\cdot s} = \frac{1}{j-1}\sum_{r=2}^{j}(r-1)\,\Delta m_{r\cdot s}, \quad j \geqslant 2, \tag{8}$$

which shows that $\delta_{j\cdot s}$ is a time-weighted average of local increments, giving more weight to recent $\Delta m_{r\cdot s}$. This provides a dynamic measure of semantic progress, emphasizing how the trajectory evolves over time.

Definition 3.4 (ERA, denoted by $\Delta_M^{(2)}$, as the *average change of historical-baseline deviation*)**.** To

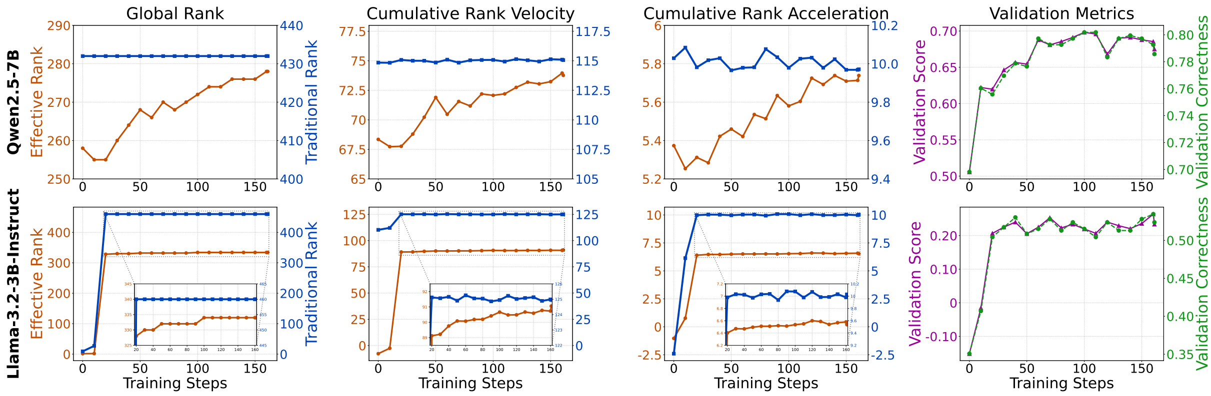


Figure 3: Visualization of dataset-level metrics during GRPO post-training. The figure compares traditional rank-based metrics with our effective-rank-based metrics. Also shown are the validation score and sample correctness, both averaged over the validation dataset.

capture whether the refinement dynamics are *accelerating* or *saturating*, we define the *average change of historical-baseline deviation*:

$$\Delta_M^{(2)} := \frac{1}{K-2} \sum_{j=3}^{K} \left[\delta_{j\cdot s} - \delta_{(j-1)\cdot s}\right]. \tag{9}$$

Interpretation. We specialize ERA $:= \Delta_{\text{ER}}^{(2)}$. Positive ERA indicates that the semantic refinement reflected by $\delta_{j\cdot s}$ is strengthening over time (accelerating refinement), while negative ERA suggests that refinement is weakening, consistent with stabilization or saturation in a narrow semantic region. In later sections, we leverage this stability-oriented signal to modulate training incentives in a principled and adaptive manner.

3.3 Scaling Properties of Semantic-Space Dynamics

In Sec. 3.1 and Sec. 3.2, we defined ER, ERV, and ERA on the hidden-state trajectory of a single response, i.e., in the model's semantic space. We now study how these representational dynamics scale at two levels: (i) **dataset-level** scaling as the number of responses N grows, and (ii) **trajectory-level** scaling as the reasoning length T increases. The key observation is that both settings can be analyzed through the same lens: the effective number of approximately orthogonal semantic directions, denoted by k. The following proposition provides a unified framework.

Proposition 3.5. *Assume a hidden-state matrix has effective support on k approximately orthogonal directions. Then ER and its corresponding ERV are upper-bounded linearly by k: ER $= \mathcal{O}(k)$ and $\Delta_{ER}^{(1)} = \mathcal{O}(k)$. In contrast, the corresponding ERA does not scale with k: $\Delta_{ER}^{(2)} = \mathcal{O}(1)$.*

Remark 3.6. Prop. 3.5 admits two interpretations, depending on what the rows of $\mathbf{Z}$ represent. **Dataset level:** with $\mathbf{Z} = \bar{\mathbf{Z}} \in \mathbb{R}^{N\times D}$ (rows are response embeddings), k measures the number of distinct semantic modes covered by the dataset, so ER/ERV grow with diversity of reasoning across questions while ERA stays $\mathcal{O}(1)$. **Response level:** with $\mathbf{Z} \in \mathbb{R}^{T\times D}$ (rows are step-wise hidden states along one trace), k measures how many reasoning steps add non-redundant semantic directions for a given question, so ER/ERV indicates novelty of reasoning within a response and $\mathcal{O}(1)$ ERA indicates non-saturating refinement.

4 Empirical Evidence for Decoupling in Semantic Space

We empirically study how exploration and exploitation evolve *in the hidden-state semantic space* during RLVR training. Concretely, we track both conventional rank and our semantic-space metrics (ER, ERV, ERA) throughout GRPO post-training. We experiment with Qwen (Hui et al., 2024) and Llama (Grattafiori et al., 2024) under the GRPO paradigm (Shao et al., 2024), using the training configuration of Zeng et al. (2025): 8k hard MATH problems (level 3-5), each paired with a verifiable reference answer. Complementary theoretical analysis is available in Appendix. E.1.

4.1 Analysis of Response-Level Metrics

At each training step, we compute response-level metrics on the current batch (Overall / Correct / Incorrect) as shown in Fig. 2, with additional anal-

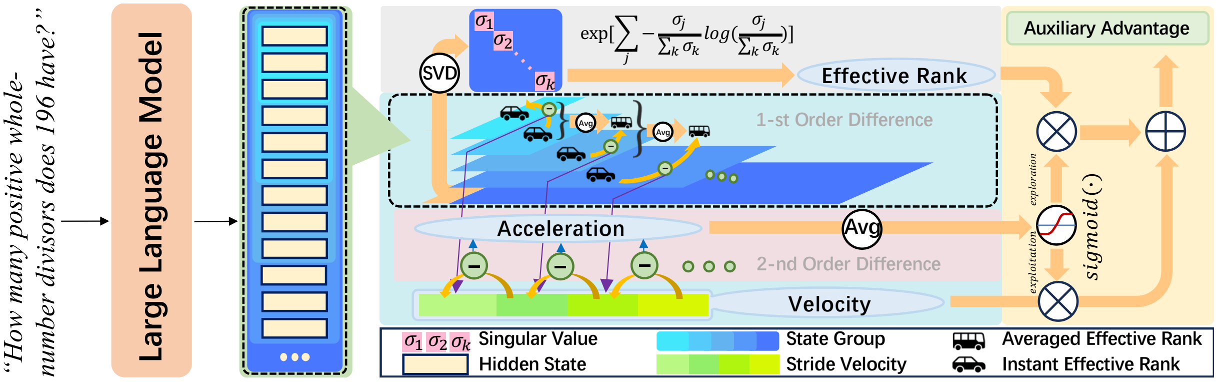


Figure 4: Overview of Velocity-Exploiting Rank-Learning. Exploration is quantified by computing the ER of the prefix hidden states via SVD, while exploitation is captured through EMA-smoothed first-order difference (ERV) on per-step prefix hidden-state matrices and extended to second-order difference (ERA). Finally, exploration and exploitation are adaptively integrated to derive the auxiliary advantage.

yses deferred to Appendix. G.1.

ER varies by model, while ERV increases consistently across models. Although RL consistently improves task performance, it induces different exploration patterns across base models in semantic space. As shown in the first column of Fig. 2, Qwen exhibits an increasing ER, suggesting broader coverage over semantic directions, whereas Llama shows a decreasing ER, consistent with a more concentrated trajectory. In contrast, ERV (second column) exhibits a consistent upward trend for both models, indicating that RL fine-tuning reliably strengthens *semantic refinement dynamics*. Concretely, representations evolve in a way that increasingly outpaces their historical trend, even when the models differ in their exploration breadth.

ERA is more strongly aligned with outcome-relevant refinement than ER or ERV alone. We further compare correct vs. incorrect trajectories in Fig. 2. Interestingly, incorrect samples often exhibit higher ER and ERV than correct ones, suggesting that larger semantic breadth (high ER) or faster growth relative to the past (high ERV) can reflect *unproductive expansion* that drifts away from a correct solution. In contrast, ERA is consistently higher for correct trajectories, indicating that correct reasoning is characterized not merely by being “large” or “fast”, but by an *increasingly coherent refinement* where the refinement dynamics strengthen over time. This suggests that ERA is a more reliable indicator of robust reasoning than zero- or first-order quantities of the ER dynamics.

4.2 Analysis of Dataset-level Metrics

Following Sec. 2, we extend the analysis from individual trajectories to the validation set by constructing the dataset hidden-state matrix $\overline{\mathbf{Z}}$. We then track the zero-, first-, and second-order dynamics of $\mathrm{erank}(\overline{\mathbf{Z}})$ during training (Fig. 3), with additional studies in Appendix. G.2.

Policy improvement correlates with expanding dataset-level semantic diversity. Across training, we observe that validation performance (accuracy and score) co-evolves with dataset-level ER dynamics. As optimization proceeds, $\mathrm{erank}(\overline{\mathbf{Z}})$ and its temporal differences tend to increase, suggesting that the learned policy occupies a richer set of semantic directions across the same problem set. The upward trends of ERV and ERA further indicate that the policy becomes progressively more effective at refining and expanding this dataset-level semantic space toward correct solutions.

ER reveals refinement even when conventional rank plateaus. In late-stage training, conventional rank may plateau, suggesting that the number of linearly independent directions in $\overline{\mathbf{Z}}$ no longer increases. However, ER can continue to rise, revealing a subtler improvement: the singular spectrum becomes more balanced, meaning the model utilizes existing semantic directions more uniformly rather than concentrating energy on a few dominant ones. This reduced redundancy indicates refinement within the established semantic space, beyond what discrete rank alone can capture.

5 Velocity-Exploiting Rank-Learning

Building on Sec. 3 and the empirical findings in Sec. 4, we propose Velocity-Exploiting Rank Learning (VERL), an advantage-shaping approach that injects *semantic-space* signals derived from hidden-state dynamics into standard RL objec-

Table 1: Performance comparison of models on mathematical reasoning benchmarks (Pass@1). "+ GRPO" and "+ PPO" denote RL fine-tuning using the GRPO and PPO frameworks, respectively. "w/ VERL" indicates incorporating our VERL into the corresponding RL algorithm. Δ represents the performance contrast between original RL method and its VERL variant. See Appendix. G.3 for full details.

Model	AIME24	AIME25	AMC23	AMC24	ASDiv	Carp_En	CMATH	Gaokao 2024_I	Gaokao 2024_Mix	Gaokao MathCloze	GSM8K	MAWPS	Olympiad Bench	SVAMP	TabMWP	Avg.
Llama-3.2-3B-Instruct	0.0	0.0	25.0	11.1	74.6	26.5	10.2	14.3	14.3	6.8	66.6	86.9	12.7	74.1	41.4	31.0
+ GRPO	3.3	0.0	27.5	8.9	88.8	45.0	28.3	21.4	20.9	23.7	80.7	96.0	16.7	87.7	71.7	41.4
+ GRPO w/ VERL	13.3	6.7	25.0	11.1	89.3	45.4	46.2	14.3	22.0	22.9	81.7	96.0	17.6	87.8	72.3	43.4
Δ_{GRPO}	+10.0	+6.7	-2.5	+2.2	+0.5	+0.4	+17.9	-7.1	+1.1	-0.8	+1.0	+0.0	+0.9	+0.1	+0.6	+2.0
+ PPO	10.0	3.3	22.5	13.3	87.9	46.4	21.2	7.1	16.5	20.3	81.4	95.5	17.8	86.8	71.0	40.1
+ PPO w/ VERL	10.0	3.3	25.0	11.1	88.7	46.0	30.7	14.3	19.8	27.1	82.9	95.7	17.3	85.8	71.3	41.9
Δ_{PPO}	+0.0	+0.0	+2.5	-2.2	+0.8	-0.4	+9.5	+7.2	+3.3	+6.8	+1.5	+0.2	-0.5	-1.0	+0.3	+1.9
Qwen2.5-7B	6.7	0.0	45.0	15.6	91.4	55.8	86.7	42.9	33.0	49.2	85.8	95.4	25.8	88.5	82.8	53.6
+ GRPO	10.0	6.7	55.0	26.7	94.8	60.2	91.7	14.3	34.1	64.4	90.2	97.6	36.1	92.8	91.3	57.7
+ GRPO w/ VERL	13.3	10.0	50.0	28.9	95.0	60.8	90.7	35.7	35.2	69.5	89.2	97.7	35.4	92.9	91.9	59.8
Δ_{GRPO}	+3.3	+3.3	-5.0	+2.2	+0.2	+0.6	-1.0	+21.4	+1.1	+5.1	-1.0	+0.1	-0.7	+0.1	+0.6	+2.1
+ PPO	6.7	3.3	50.0	33.3	94.9	59.6	89.8	28.6	31.9	63.6	89.1	97.3	36.1	92.8	90.8	57.9
+ PPO w/ VERL	10.0	6.7	52.5	33.3	94.8	60.0	90.3	28.6	34.1	66.9	90.2	97.8	36.1	92.5	90.6	59.0
Δ_{PPO}	+3.3	+3.3	+2.5	+0.0	-0.1	+0.4	+0.5	+0.0	+2.2	+3.3	+1.1	+0.5	+0.0	-0.3	-0.2	+1.1

Table 2: Performance comparison of models on mathematical reasoning benchmarks (Pass@k). "+ GRPO" and "+ PPO" denote RL fine-tuning using the GRPO and PPO frameworks, respectively. "w/ VERL" indicates incorporating our VERL into the corresponding RL algorithm. Δ represents the performance contrast between original RL method and its VERL variant. See Appendix. G.4 for full details.

Model	MATH500 (Pass@16)	AMC23 (Pass@128)	AMC24 (Pass@128)	AIME24 (Pass@256)	AIME25 (Pass@256)	Avg.
Llama-3.2-3B-Instruct	79.8	93.5	51.5	40.0	30.0	58.96
+ GRPO	80.2	95.4	60.6	40.0	30.0	61.24
+ GRPO w/ VERL	80.6	95.7	59.0	50.0	36.7	64.40
Δ_{GRPO}	+0.4	+0.3	−1.6	+10.0	+6.7	+3.16
+ PPO	82.2	94.5	57.0	46.7	36.7	63.42
+ PPO w/ VERL	82.4	94.7	57.8	46.7	40.0	64.32
Δ_{PPO}	+0.2	+0.2	+0.8	+0.0	+3.3	+0.90
Qwen2.5-7B	90.6	98.4	73.7	60.0	60.0	76.54
+ GRPO	90.8	97.8	78.3	56.7	50.0	74.72
+ GRPO w/ VERL	91.4	98.3	79.0	63.3	60.0	78.40
Δ_{GRPO}	+0.6	+0.5	+0.7	+6.6	+10.0	+3.68
+ PPO	91.2	98.6	74.3	53.3	56.7	74.82
+ PPO w/ VERL	91.4	98.0	74.4	56.7	66.7	77.44
Δ_{PPO}	+0.2	−0.6	+0.1	+3.4	+10.0	+2.62

tives. Conventional RLVR optimizes rewards in the token-level action space, but does not explicitly regulate how reasoning trajectories evolve in representation space. This can yield inefficient updates, e.g., unproductive semantic expansion or premature collapse into narrow reasoning patterns. VERL addresses this by constructing an auxiliary advantage from ER, ERV, and ERA, encouraging both broader semantic coverage and productive refinement in a unified and stable manner.

5.1 Normalized Deviations in Semantic Space

We compute three scalar metrics per trajectory, $\{m_0, m_1, m_2\}$, corresponding to ER (breadth), ERV (first-order refinement dynamics), and ERA (second-order refinement dynamics). To obtain a stable, scale-invariant control signal across training, we normalize each metric by its recent trend using an exponential moving average (EMA) $\bar{\mu}_k$. We define the relative deviation:

$$d_k := (m_k - \bar{\mu}_k)/|\bar{\mu}_k| \quad k \in \{0, 1, 2\}. \qquad (10)$$

Intuitively, d_k measures how the current trajectory deviates from the policy's recent behavior in semantic space.

5.2 Adaptive Balancing via ERA as a Meta-Control Signal

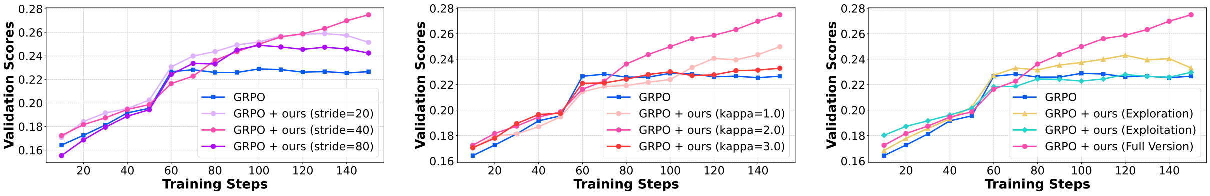
Sec. 3.1–3.2 motivates ER (M_0) as a proxy for exploration breadth in semantic space and ERV (M_1) as a proxy for refinement dynamics (exploitation). Empirically, these two quantities are nearly uncorrelated (Fig. 1b), suggesting they can be jointly optimized rather than traded off. We therefore combine M_0 and M_1 into a single auxiliary signal, and use ERA (M_2) as a meta-control variable to adapt their relative emphasis.

The key intuition is that ERA captures the *change* of refinement dynamics. When refinement accelerates strongly relative to its EMA trend ($d_2 \gg 0$), the policy may be at risk of collapsing into overly confident, narrow patterns that generalize poorly. When refinement decelerates or saturates ($d_2 \leqslant 0$), strengthening exploitation becomes more important to consolidate useful progress. This motivates an ERA-controlled interpolation between exploration- and exploitation-oriented weights.

We define two basis weight vectors, $\mathbf{w}_{\text{explore}} = [1, 0]$ for M_0 and $\mathbf{w}_{\text{exploit}} = [0, 1]$ for M_1. The

Table 3: Pass@1 performance under different β settings. "Adaptive β" denotes $\beta := \text{sigmoid}(d_2)$. Accuracy columns are percentages, whereas Score Avg reports the average validation score. **Boldface** indicates the best result.

Training Strategy	Score Avg	In Domain			Out of Domain				Hard Problems				
		MATH	MATH500	Avg	Gaokao	CN Middle School	CMATH	Avg	AIME24	AIME25	AMC23	AMC24	Avg
GRPO	0.36	**51.4**	46.2	48.80	**23.7**	28.7	28.3	26.90	3.3	0.0	**27.5**	8.9	9.93
GRPO+VERL ($\beta = 0.5$)	**0.38**	51.2	47.2	49.20	21.2	36.6	38.7	32.17	10.0	0.0	**27.5**	8.9	11.60
GRPO+VERL (Adaptive β)	**0.38**	50.9	**51.2**	**51.05**	22.9	**38.6**	**46.2**	**35.90**	**13.3**	**6.7**	20.0	**11.1**	**12.78**



(a) Different Strides (s). (b) Different advantage clip (κ). (c) Different shaping signals.

Figure 5: Hyperparameter ablations for Llama-3.2-3B-Instruct. Panels (a) and (b) show that VERL performs best with stride $s = 40$ and clipping factor $\kappa = 2$, respectively. Panel (c) shows that using either the exploration or exploitation term alone is suboptimal, supporting the effectiveness of the combined signal.

dynamic weight is:

$$\mathbf{w}_{\text{dyn}} := \beta\, \mathbf{w}_{\text{explore}} + (1 - \beta)\, \mathbf{w}_{\text{exploit}}, \quad (11)$$

where $\beta := \sigma(d_2)$ and $\sigma(\cdot)$ is the sigmoid. We then define the auxiliary trajectory-level signal:

$$\Phi := w_{\text{dyn},0} \cdot \tanh(d_0) + w_{\text{dyn},1} \cdot \tanh(d_1), \quad (12)$$

where $\tanh$ bounds the auxiliary signal to improve stability. Intuitively, Φ rewards trajectories whose semantic breadth or refinement exceeds recent trends, while using d_2 to adaptively mitigate over-collapse and stagnation.

5.3 Advantage Shaping with Auxiliary Semantic-Space Signals

We shape the learning signal by adding a clipped, non-negative bonus derived from Φ. Let A_t denote the original advantage (e.g., from GRPO or PPO with GAE), and let Φ_i be the trajectory-level auxiliary signal in Sec. 5.2. We use the refined advantage:

$$\hat{A}_t = A_t + \min(\text{ReLU}(\Phi_i),\ |A_t|/\kappa), \quad (13)$$

which preserves the sign of A_t while bounding the shaping magnitude for stability. For GRPO, Φ_i is applied once per trajectory; for PPO+GAE, the same Φ_i is broadcast to all tokens in the trajectory.

6 Experiments

6.1 Experiment Settings

(i) **Datasets.** We use the same training and evaluation datasets as in Sec. 4. *(ii)* **Reward.** We employ a rule-based reward that checks mathematical correctness and `\boxed{}` formatting. A correct answer receives $+1.0$ if properly formatted and $+0.5$ otherwise; an incorrect answer receives -0.5 if formatted and -1.0 otherwise. *(iii)* **Training.** We build on VERL (Sheng et al., 2025) with vLLM (Kwon et al., 2023). We use batch size 48, generate 4 rollouts per prompt for GRPO and 1 rollout for PPO, and set the maximum generation length $L_{\max} = 1536$.

6.2 Main Results

Generalization across benchmarks. Tab. 1 summarizes Pass@1 results on 15 mathematical reasoning benchmarks (full tables in Appendix. G.3). Across both easy (e.g., ASDiv) and hard benchmarks (e.g., OlympiadBench), VERL consistently improves over the corresponding RL baselines. Gains are most pronounced on multi-step reasoning benchmarks, including **+21.4** on Gaokao 2024_I with Qwen2.5-7B and **+10.0** on AIME24 with Llama-3.2-3B-Instruct.

Generalization across RL algorithms and base models. VERL is plug-and-play: it can be paired with both GRPO and PPO, and improves average performance for both Llama and Qwen models in Tab. 1. This indicates that semantic-space advantage shaping transfers across optimization algorithms and base-model families.

Improving both exploration and exploitation proxies. Tab. 2 reports Pass@k under diverse decoding budgets (Appendix. G.4). Pass@1 is commonly used as a proxy for strong single-shot performance, whereas Pass@k reflects the ability to succeed given a larger search budget. We observe larger gains on Pass@k than on Pass@1, especially on harder benchmarks, suggesting that VERL improves not only single-trajectory refinement but also the coverage of successful reasoning trajectories under increased sampling. See Sec. H for qualitative analysis.

Performance Degradation. As shown in Tab. 1, VERL's degradations are localized rather than systematic. Under GRPO, the main drops are Llama on Gaokao 2024_I (-7.1) and Qwen on AMC23/CMATH (-5.0/-1.0), yet the corresponding average Pass@1 still improves by +2.0 and +2.1 points, respectively. These drops are also offset by large gains on challenging benchmarks, such as AIME24 (+10.0) and CMATH (+17.9) for Llama, and Gaokao 2024_I (+21.4) for Qwen. Moreover, the Llama drop on Gaokao 2024_I is optimizer-dependent, as VERL improves the same benchmark by +7.2 under PPO. Overall, the occasional benchmark-level fluctuations are outweighed by consistent aggregate gains.

6.3 Ablations and Analysis

We ablate key components and hyperparameters of VERL, including the role of ERA, the stride s for temporal metrics, the advantage clip κ, and the composition of the shaping signal Φ.

Effectiveness of ERA. Tab. 3 shows that adding semantic-space shaping improves over GRPO. Moreover, adaptive mixing ($\beta = \sigma(d_2)$) outperforms fixed mixing ($\beta = 0.5$), indicating that ERA provides a useful meta-control signal for adjusting when to emphasize breadth (ER) versus refinement (ERV). This suggests that VERL improves performance on OOD and challenging tasks.

Temporal stride s. The stride s controls the temporal granularity for computing ERV and ERA. As shown in Fig. 5a, improvements are robust across a range of s, suggesting that the signal is not overly sensitive to sampling frequency. We use $s = 40$ in subsequent experiments as it yields the best validation reward while avoiding noisy token-level fluctuations.

Advantage clip κ. The clip factor κ bounds the shaping bonus so it complements, rather than dominates, the original advantage. As shown in Fig. 5b, VERL consistently improves performance across all tested κ, indicating robust behavior to this hyperparameter. The best performance is achieved at $\kappa = 2$, which provides a sufficiently strong yet well-proportioned shaping signal.

Shaping signal Φ. Fig. 5c compares variants of Eq. 12. Using only the exploration term improves coverage but under-utilizes high-reward trajectories, leading to earlier bottlenecks. Using only the exploitation term yields faster initial gains but plateaus due to insufficient breadth. Combining both terms achieves the most stable training and the best final performance, consistent with our goal of improving both capacities in semantic space.

7 Conclusion

We argue that the exploration–exploitation trade-off in RLVR for LLM reasoning is largely a measurement artifact of token-level action statistics. Moving to hidden-state semantic space, we propose ER, ERV, and ERA to capture semantic breadth and refinement dynamics, and empirically find near-zero correlation between exploration and exploitation. Leveraging ERA as a meta-control signal, VERL performs semantic-space advantage shaping and consistently improves both in-domain performance and out-of-domain generalization on challenging reasoning benchmarks.

Limitations

We have not systematically studied how training-data cleaning or curation affects VERL. While VERL shows clear advantages on math-focused training data with verifiable answers, extending these gains to other reasoning training corpora remains an open direction for future work. Furthermore, we have not addressed the computational costs associated with scaling VERL to larger, more diverse training datasets.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant Nos. 92467204 and 62472249), the Shenzhen Science and Technology Program (Grant No. KJZD20240903102300001), the Shenzhen Science and Technology Program (Grant No. JCYJ20250604145014018), and the Natural Science Foundation of Top Talent of SZTU (Grant No. GDRC202413). The authors used AI assistants only for grammar and spelling correction; all scientific content was written and verified by the authors.

References

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. 2025. The unreasonable effectiveness of entropy minimization in llm reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS 2025 Poster.

Shelly Bensal, Umar Jamil, Christopher Bryant, Melisa Russak, Kiran Kamble, Dmytro Mozolevskyi, Muayad Ali, and Waseem AlShikh. 2025. Reflect, retry, reward: Self-improving llms via reinforcement learning. *arXiv preprint arXiv:2505.24726*.

Yao Chen, Yilong Chen, Yinqi Yang, Junyuan Shang, Zhenyu Zhang, Zefeng Zhang, Shuaiyi Nie, Shuohuan Wang, Yu Sun, Hua Wu, and 1 others. 2026. Sparse growing transformer: Training-time sparse depth allocation via progressive attention looping. *arXiv preprint arXiv:2603.23998*.

Yao Chen, Jiawei Sheng, Wenyuan Zhang, and Tingwen Liu. 2025a. Improving reasoning capabilities in small models through mixture-of-layers distillation with stepwise attention on key information. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 4952–4971.

Zhipeng Chen, Xiaobo Qin, Youbin Wu, Yue Ling, Qinghao Ye, Wayne Xin Zhao, and Guang Shi. 2025b. Pass@ k training for adaptively balancing exploration and exploitation of large reasoning models. *arXiv preprint arXiv:2508.10751*.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Xin Zhao, Zhenliang Zhang, and Furu Wei. 2026. Reasoning with exploration: An entropy perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30377–30385.

Zhengxiang Cheng, Dongping Chen, Mingyang Fu, and Tianyi Zhou. 2025. Lc-r1: Optimizing length compression in large reasoning model. In *ICML 2025 Affinity Workshop: New In ML*. Poster.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, and 1 others. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

Runpeng Dai, Linfeng Song, Haolin Liu, Zhenwen Liang, Dian Yu, Haitao Mi, Zhaopeng Tu, Rui Liu, Tong Zheng, Hongtu Zhu, and Dong Yu. 2026. CDE: Curiosity-driven exploration for efficient reinforcement learning in large language models. In *The Fourteenth International Conference on Learning Representations*.

Mehul Damani, Isha Puri, Stewart Slocum, Idan Shenfeld, Leshem Choshen, Yoon Kim, and Jacob Andreas. 2026. Beyond binary rewards: Training LMs to reason about their uncertainty. In *The Fourteenth International Conference on Learning Representations*.

Jia Deng, Jie Chen, Zhipeng Chen, Daixuan Cheng, Fei Bai, Beichen Zhang, Yinqian Min, Yanzipeng Gao, Wayne Xin Zhao, and Ji-Rong Wen. 2025a. From trial-and-error to improvement: A systematic analysis of llm exploration mechanisms in rlvr. *arXiv preprint arXiv:2508.07534*.

Wenlong Deng, Yi Ren, Danica J. Sutherland, Christos Thrampoulidis, and Xiaoxiao Li. 2025b. Token hidden reward: Steering exploration-exploitation in grpo training. In *ICML 2025 Workshop on AI for Math (AI4MATH)*. Oral.

Tianwei Ding, Lanshan He, Weijian Ma, and Xiangdong Zhou. 2025. Musellm: Sdf generation and understanding via multi-scale tokenization with position-aware guidance. In *Proceedings of the 2025 International Conference on Multimedia Retrieval*, ICMR '25, page 192–201, New York, NY, USA. Association for Computing Machinery.

Boyan Duan, Xiao Liang, Shuai Lu, Yaoxiang Wang, Yelong Shen, Kai-Wei Chang, Ying Nian Wu, Mao Yang, Weizhu Chen, and Yeyun Gong. 2025. Gold-medal-level olympiad geometry solving with efficient heuristic auxiliary constructions. *arXiv preprint arXiv:2512.00097*.

Yu Fan, Jingwei Ni, Jakob Merane, Yang Tian, Yoan Hermstrüwer, Yinya Huang, Mubashara Akhtar, Etienne Salimbeni, Florian Geering, Oliver Dreyer, and 1 others. 2025. Lexam: Benchmarking legal reasoning on 340 law exams. *arXiv preprint arXiv:2505.12864*.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. 2025. Deep think with confidence. In *NeurIPS 2025 Workshop on ER*. Spotlight.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633–638.

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

Fanding Huang, Jingyan Jiang, Qinting Jiang, Hebei Li, Faisal Nadeem Khan, and Zhi Wang. 2025. COSMIC: clique-oriented semantic multi-space integration for robust CLIP test-time adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pages 9772–9781. Computer Vision Foundation / IEEE.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

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*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Yuxuan Jiang and Francis Ferraro. 2026a. Beyond math: Stories as a testbed for memorization-constrained reasoning in llms. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5590–5607.

Yuxuan Jiang and Francis Ferraro. 2026b. Scribe: Structured mid-level supervision for tool-using language models. *Preprint*, arXiv:2601.03555.

Yuxuan Jiang, Dawei Li, and Frank Ferraro. 2025. Drp: Distilled reasoning pruning with skill-aware step decomposition for efficient large reasoning models. *arXiv preprint arXiv:2505.13975*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O. Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. In *Conference on Language Modeling (COLM)*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2567–2577. Association for Computational Linguistics.

Yi Jing, Zijun Yao, Hongzhu Guo, Lingxu Ran, Xiaozhi Wang, Lei Hou, and Juanzi Li. 2025. LinguaLens: Towards interpreting linguistic mechanisms of large language models via sparse auto-encoder. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 28232–28251, Suzhou, China. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 1152–1157.

Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. 2021. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *Proceedings of the 9th International Conference on Learning Representations (ICLR 2021)*.

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

Timothée Lesort, Natalia Díaz Rodríguez, Jean-François Goudou, and David Filliat. 2018. State representation learning for control: An overview. *Neural networks : the official journal of the International Neural Network Society*, 108:379–392.

Haitao Li, You Chen, Qingyao Ai, Yueyue Wu, Ruizhe Zhang, and Yiqun Liu. 2024. Lexeval: A comprehensive chinese legal benchmark for evaluating large language models. *Advances in Neural Information Processing Systems*, 37:25061–25094.

Jie Li, Fuyong Zhao, Panfeng Chen, Jiafu Xie, Xiangrui Zhang, Hui Li, Mei Chen, Yanhao Wang, and Ming Zhu. 2025a. An astronomical question answering dataset for evaluating large language models. *Scientific Data*, 12(1):447.

Pengyi Li, Matvey Skripkin, Alexander Zubrey, Andrey Kuznetsov, and Ivan Oseledets. 2025b. Confidence is all you need: Few-shot rl fine-tuning of language models. *arXiv preprint arXiv:2506.06395*.

Zhong-Zhi Li, Xiao Liang, Zihao Tang, Lei Ji, Peijie Wang, Haotian Xu, Haizhen Huang, Weiwei Deng, Yeyun Gong, Zhijiang Guo, and 1 others. 2025c. Tl; dr: Too long, do re-weighting for efficient llm reasoning compression. *arXiv preprint arXiv:2506.02678*.

Xiao Liang, Zhong-Zhi Li, Yeyun Gong, Yang Wang, Hengyuan Zhang, Yelong Shen, Ying Nian Wu, and Weizhu Chen. 2025. Sws: Self-aware weakness-driven problem synthesis in reinforcement learning for llm reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS 2025 Poster.

Xiao Liang, Zhong-Zhi Li, Yeyun Gong, yelong shen, Ying Nian Wu, Zhijiang Guo, and Weizhu Chen. 2026. Beyond pass@ 1: Self-play with variational problem synthesis sustains RLVR. In *The Fourteenth International Conference on Learning Representations*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3622–3628. International Joint Conferences on Artificial Intelligence Organization. Main track.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2023. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *ICLR*.

Clare Lyle, Mark Rowland, and Will Dabney. 2022. Understanding and preventing capacity loss in reinforcement learning. In *Proceedings of the 10th International Conference on Learning Representations (ICLR 2022)*.

Weijian Ma, Ruoxin Chen, Keyue Zhang, Shuang Wu, and Shouhong Ding. 2025. Instruct where the model fails: Generative data augmentation via guided self-contrastive fine-tuning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(6):5991–5999.

Weijian Ma, Shizhao Sun, Ruiyu Wang, and Jiang Bian. Cadmorph: Geometry-driven parametric cad editing via a plan–generate–verify loop. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Weijian Ma, Shizhao Sun, Tianyu Yu, Ruiyu Wang, Tat-Seng Chua, and Jiang Bian. 2026. Thinking with blueprints: Assisting vision-language models in spatial reasoning via structured object representation. *Preprint*, arXiv:2601.01984.

Jacob A. Matthews, John R. Starr, and Marten van Schijndel. 2024. Semantics or spelling? probing contextual word embeddings with orthographic noise. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4445–4462, Bangkok, Thailand. Association for Computational Linguistics.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on Health, Inference, and Learning, CHIL 2022, 7-8 April 2022, Virtual Event*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

Ziqing Qiao, Yongheng Deng, Jiali Zeng, Dong Wang, Lai Wei, Guanbo Wang, Fandong Meng, Jie Zhou, Ju Ren, and Yaoxue Zhang. 2025. ConCISE: Confidence-guided compression in step-by-step efficient reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8010–8029, Suzhou, China. Association for Computational Linguistics.

Corrado Rainone, Tim Bakker, and Roland Memisevic. 2025. Replacing thinking with tool usage enables reasoning in small language models. In *ICML 2025 Workshop on Test-Time Adaptation: Putting Updates to the Test (PUT)*. Poster.

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

Olivier Roy and Martin Vetterli. 2007. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pages 606–610. IEEE.

Hassan Sajjad, Nadir Durrani, Fahim Dalvi, Firoj Alam, Abdul Khan, and Jia Xu. 2022. Analyzing encoded concepts in transformer language models. In *Proceedings of the 2022 Conference of the North American chapter of the Association for Computational Linguistics: Human language technologies*, pages 3082–3101.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. 2016. High-dimensional continuous control using generalized advantage estimation. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

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

Giovanni Servedio, Alessandro De Bellis, Dario Di Palma, Vito Walter Anelli, and Tommaso Di Noia. 2025. Are the hidden states hiding something? testing the limits of factuality-encoding capabilities in llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 6089–6104. Association for Computational Linguistics.

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*.

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

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. 2025. Layer by layer: Uncovering hidden representations in language models. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

Yuxi Sun, Aoqi Zuo, Wei Gao, and Jing Ma. 2025. Causalabstain: Enhancing multilingual llms with causal reasoning for trustworthy abstention. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 14060–14076.

Yuxi Sun, Aoqi Zuo, Haotian Xie, Wei Gao, Mingming Gong, and Jing Ma. 2026. Fact-e: Causality-inspired evaluation for trustworthy chain-of-thought reasoning. *Preprint*, arXiv:2604.10693.

Shi-Yu Tian, Zhi Zhou, Wei Dong, Kun-Yang Yu, Ming Yang, Zi-Jian Cheng, Lan-Zhe Guo, and Yu-Feng Li. 2025. Tabularmath: Understanding math reasoning over tables with large language models. *arXiv preprint arXiv:2505.19563*.

Lucrezia Valeriani, Diego Doimo, Francesca Cuturello, Alessandro Laio, Alessio Ansuini, and Alberto Cazzaniga. 2023. The geometry of hidden representations of large transformer models. *Advances in Neural Information Processing Systems*, 36:51234–51252.

Chengbing Wang, Yang Zhang, Wenjie Wang, Xiaoyan Zhao, Fuli Feng, Xiangnan He, and Tat-Seng Chua. 2026. Think-while-generating: On-the-fly reasoning for personalized long-form generation. In *The Fourteenth International Conference on Learning Representations*.

Han Wang, Erfan Miahi, Martha White, Marlos C. Machado, Zaheer Abbas, Raksha Kumaraswamy, Vincent Liu, and Adam White. 2024a. Investigating the properties of neural network representations in reinforcement learning. *Artificial Intelligence*, page 104100.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS 2025 Poster.

Wenqiang Wang and Yangshijie Zhang. 2025. Incomplete in-context learning. *arXiv preprint arXiv:2505.07251*.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R. Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. 2024b. SciBench: Evaluating College-Level Scientific Problem-Solving Abilities of Large Language Models. In *Proceedings of the Forty-First International Conference on Machine Learning*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024c. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

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

Tianwen Wei, Jian Luan, Wei Liu, Shuang Dong, and Bin Wang. 2023. Cmath: Can your language model pass chinese elementary school math test? *arXiv preprint arXiv:2306.16636*.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, Jiang Bian, and Mao Yang. 2026. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base LLMs. In *The Fourteenth International Conference on Learning Representations*.

Ningning Xu, Yuxuan Jiang, and Shubhashis Roy Dipta. 2025. Learning how to use tools, not just when: Pattern-aware tool-integrated reasoning. In *The 5th Workshop on Mathematical Reasoning and AI at NeurIPS 2025*.

Shijia Xu, Yu Wang, Xiaolong Jia, Zhou Wu, Kai Liu, and April Xiaowen Dong. 2026a. Rcbsf: A multi-agent framework for automated contract revision via stackelberg game. *Preprint*, arXiv:2604.10740.

Shijia Xu, Zhou Wu, Xiaolong Jia, Yu Wang, Kai Liu, and April Xiaowen Dong. 2026b. Self-correcting rag: Enhancing faithfulness via mmkp context selection and nli-guided mcts. *Preprint*, arXiv:2604.10734.

Chao Xue, Yao Wang, Mengqiao Liu, Di Liang, Xingsheng Han, Peiyang Liu, Xianjie Wu, Chenyao Lu, Lei Jiang, Yu Lu, Haibo Shi, Shuang Liang, Minlong Peng, and Flora D. Salim. 2026a. Reason only when needed: Efficient generative reward modeling via model-internal uncertainty. *Preprint*, arXiv:2604.10072.

Chao Xue, Yao Wang, Mengqiao Liu, Di Liang, Xingsheng Han, Peiyang Liu, Xianjie Wu, Chenyao Lu, Lei Jiang, Yu Lu, Haibo Shi, Shuang Liang, Minlong Peng, and Flora D. Salim. 2026b. Why supervised fine-tuning fails to learn: A systematic study of incomplete learning in large language models. *Preprint*, arXiv:2604.10079.

Hang Yan, Fangzhi Xu, Rongman Xu, Yifei Li, Jian Zhang, Haoran Luo, Xiaobao Wu, Luu Anh Tuan, Haiteng Zhao, Qika Lin, and 1 others. 2025. Mur: Momentum uncertainty guided reasoning for large language models. *arXiv preprint arXiv:2507.14958*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 2 others. 2025a. Dapo: An open-source llm reinforcement learning system at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS 2025 Poster.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yiyao Yu, Yuxiang Zhang, Dongdong Zhang, Xiao Liang, Hengyuan Zhang, Xingxing Zhang, Mahmoud Khademi, Hany Hassan Awadalla, Junjie Wang, Yujiu Yang, and 1 others. 2025b. Chain-of-reasoning: Towards unified mathematical reasoning in large language models via a multi-paradigm perspective. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24914–24937.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun MA, and Junxian He. 2025. SimpleRL-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. In *Second Conference on Language Modeling*.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. 2025a. Reasoning models know when they're right: Probing hidden states for self-verification. In *Second Conference on Language Modeling (COLM)*.

Beichen Zhang, Kun Zhou, Xilin Wei, Xin Zhao, Jing Sha, Shijin Wang, and Ji-Rong Wen. 2023. Evaluating and improving tool-augmented computation-intensive math reasoning. *Advances in Neural Information Processing Systems*, 36:23570–23589.

Duzhen Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhijiang Guo, Le Song, and Cheng-Lin Liu. 2026. From System 1 to System 2: A Survey of Reasoning Large Language Models . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 48(03):3335–3354.

Hengyuan Zhang, Shiping Yang, Xiao Liang, Chenming Shang, Yuxuan Jiang, Chaofan Tao, Jing Xiong, Hayden Kwok-Hay So, Ruobing Xie, Angel X Chang, and 1 others. 2025b. Find your optimal teacher: Personalized data synthesis via router-guided multi-teacher distillation. *arXiv preprint arXiv:2510.10925*.

Xiaofan Zhou and Lu Cheng. 2025. Robust uncertainty quantification for self-evolving large language models via continual domain pretraining. *arXiv preprint arXiv:2510.22931*.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, Biqing Qi, Youbang Sun, Zhiyuan Ma, Lifan Yuan, Ning Ding, and Bowen Zhou. 2025. Ttrl: Test-time reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS 2025 Poster.

Contents of Appendix

A Notations

Symbol	Description
π	Large language model policy
θ	The parameter of large language model policy
ϕ	The parameter corresponding to the optimal policy
t	Time step
T	Response length
L_{max}	The maximum length of model's output
x	Prompt
$\mathcal{P}_x$	Distribution of prompts
y_t	The t-th step (token) of the model's response
$y_{i:j}$	Sequence of reasoning steps from i to j
D	Feature dimension of hidden states
N	The size of the dataset
z_t	Hidden state corresponding to the t-th step of the output token
$\bar{z}_i$	Single vector representation for the i-th response by averaging its token hidden states
$\mathbf{Z}_c$	Mean-centered hidden state matrix
$\overline{\mathbf{Z}}_{1:n}$	Dataset-level hidden states matrix formed by the first n prompts
$\mathbf{Z}$	Response-level hidden states matrix
$\mathrm{SVD}(\cdot)$	The function to calculate the singular values
$\lambda_j(\cdot)$	The j-th eigenvalues of the given matrix
σ_j	The j-th singular value of matrix
p_j	The j-th normalized singular values
$\mathbf{p}$	Singular value distribution
s	The stride for effective rank velocity calculation
$\mathrm{rank}(\cdot)/\mathrm{erank}(\cdot)$	Conventional rank/Effective rank function
$\Delta_M^{(i)}$	The i-order temporal difference for metric M
$\mathcal{M}$	Set of metrics derived from the hidden states
M_i	The i-order temporal difference of ER, exactly the different metrics
$\bar{\mu}_k$	Exponential moving average of metric M_k over training steps
m_t	Value of metric M computed on the token sub-sequence from the start to position t
$\delta_n^{(i)}$	Instantaneous i-order difference for step n
$\epsilon_{\text{high/low}}$	Hyperparameter for the upper/lower bound used for clipping
ϵ	Small constant for numerical stability
$\mathbf{w}_{\text{explore/exploit/dyn}}$	Exploration-focused profile/Exploitation-focused profile/Dynamic-weighted profile
$w_{dyn,i}$	The i-th scalar of $\mathbf{w}_{\text{dynamic}}$
r_j	Reward of the j-th response
$A_{i,t}$	Group-relative advantage for the t-th token in the i-th response in group
A	The original advantage estimation
$\hat{A}$	The reshaped advantage value
Φ	Auxiliary advantage
β	Interpolation coefficient for VERL training
G	Size of sampled group in GRPO
ρ_t	Probability ratio between the current and old policies for t-step of the output
$\mathcal{L}_{\text{PPO}}(\cdot)$	The optimization objective for PPO applied to policy

B Algorithm

Algorithm 1 VERL: Training

Input: dataset $\mathcal{D}$, prompt $x \in \mathcal{D}$, $|\mathcal{D}| = N$, policy model π_θ, hidden-state dimension D and number of rollouts per prompt S.
Parameters: EMA factor γ, relative deviation stabilizing factor $\epsilon \ll 1$, RL fine-tuning stabilizing factor κ.
Initialize: Randomly initialize policy parameters π_θ, historical averages of metrics $\bar{\mu}_{\text{ER}} = \bar{\mu}_{\text{ERV}} = \bar{\mu}_{\text{ERA}} = 0$, exploration capacity profile$\mathbf{w}_{\text{explore}} = [1, 0]$, exploitation capacity profile $\mathbf{w}_{\text{exploit}} = [0, 1]$.
Output: A well-trained policy model π_θ.
repeat
  for $x \in \mathcal{D}$ **do**: // Pick a sample from dataset
    for 1 **to** S **do**: // Generate S rollouts for one sample
      $y_0 \leftarrow x$, $\mathbf{Z}_0 \leftarrow \varnothing$, $t \leftarrow 1$
      repeat // Generation process
        $y_t, \mathbf{z}_t \sim \pi_\theta(\cdot|y_{t-1})$
        $y_t \leftarrow [y_{t-1}; y_t]$ // Concatenate token sequence
        $\mathbf{Z}_t \leftarrow [\mathbf{Z}_{t-1}; \mathbf{z}_t] \in \mathbb{R}^{t \times D}$
        $\boldsymbol{\sigma}_t \leftarrow \text{SVD}(\mathbf{Z}_t)$
        $j \leftarrow |\boldsymbol{\sigma}_t|$
        $p_{j,t} \leftarrow \sigma_{j,t} / \sum_j \sigma_{j,t}$
        $\text{erank}_t \leftarrow \exp\left(-\sum_j p_{j,t} \log p_{j,t}\right)$
        If $t > 1$ **then**: $\delta_{\text{ERV},t} \leftarrow \text{erank}_t - \frac{1}{t-1} \sum_{k=1}^{t-1} \text{erank}_k$
        If $t > 2$ **then**: $\delta_{\text{ERA},t} \leftarrow \delta_{\text{ERV},t} - \delta_{\text{ERV},t-1}$
        $t \leftarrow t + 1$
      until response generation terminates; // $t-1$ is the final timestep after generation
      $A_{origin} \leftarrow$ base RL evaluating on y_{t-1}
      $m_{\text{ER}} \leftarrow \text{erank}_{t-1}$ // Calculating ER metric
      $m_{\text{ERV}} \leftarrow \frac{1}{t-2} \sum_{t=2}^{t-1} \delta_{\text{ERV}}^t$ // Calculating ERV metric
      $m_{\text{ERA}} \leftarrow \frac{1}{t-3} \sum_{t=3}^{t-1} \delta_{\text{ERA}}^t$ // Calculating ERA metric
      $\bar{\mu}_k \leftarrow \gamma \bar{\mu}_k + (1-\gamma) m_k$, $k \in \{\text{ER}, \text{ERV}, \text{ERA}\}$
      $d_k \leftarrow \frac{m_k - \bar{\mu}_k}{|\bar{\mu}_k| + \epsilon}$, $k \in \{\text{ER}, \text{ERV}, \text{ERA}\}$
      $\beta \leftarrow \text{sigmoid}(d_{\text{ERA}})$
      $\mathbf{w}_{\text{dyn}} \leftarrow \beta \mathbf{w}_{\text{explore}} + (1-\beta) \mathbf{w}_{\text{exploit}}$
      $w_{\text{dyn,ER}} \leftarrow$ the first scalar value of $\mathbf{w}_{\text{dyn}}$
      $w_{\text{dyn,ERV}} \leftarrow$ the second scalar value of $\mathbf{w}_{\text{dyn}}$
      $\Phi \leftarrow w_{\text{dyn,ER}} \tanh(d_{\text{ER}}) + w_{\text{dyn,ERV}} \tanh(d_{\text{ERV}})$
      $\hat{A} \leftarrow A_{origin} + \min\left(\text{ReLU}(\Phi), \frac{|A_{origin}|}{\kappa}\right)$
    end for
  end for
  Update θ via base RL objective with $\hat{A}$
until θ converges;

C Related Work

C.1 Reasoning with LLMs

Reasoning with LLMs has become a central topic in recent large-model research, with surveys highlighting a shift from fast pattern to more deliberate reasoning behaviors (Zhang et al., 2026). Prior work has studied mathematical reasoning and challenging specialized domains such as olympiad-level geometry solving (Yu et al., 2025b; Duan et al., 2025; Tian et al., 2025), while more recent efforts extend LLM reasoning to domain-specific and multimodal settings (Jiang and Ferraro, 2026a), including contract revision (Xu et al., 2026a), self-correcting RAG (Xu et al., 2026b), personalized long-form generation (Wang et al., 2026), structured spatial reasoning (Ma et al., 2026), CAD editing (Ma et al.), and 3D understanding (Ding et al., 2025). Related efforts also study tool-using reasoning, including pattern-aware tool-integrated reasoning (Xu et al., 2025) and structured mid-level supervision for tool-using language models (Jiang and Ferraro, 2026b). At the model level, prior studies improve reasoning capability or efficiency through task-aware data augmentation (Ma et al., 2025), sparse computation (Chen et al., 2026), distillation (Chen et al., 2025a), reasoning-oriented pruning (Jiang et al., 2025), and multi-teacher data synthesis (Zhang et al., 2025b), while another line emphasizes more reliable and better-directed reasoning via causal abstention (Sun et al., 2025), causality-inspired evaluation (Sun et al., 2026), robust uncertainty quantification for self-evolving LLMs (Zhou and Cheng, 2025), and curiosity-driven exploration (Dai et al., 2026). In contrast, we focus on reasoning dynamics during RL training and study how exploration and exploitation can be characterized and improved in semantic space.

C.2 Reinforcement Learning with Verifiable Rewards

Compared with supervised fine-tuning (Xue et al., 2026b) or in-context learning (Wang and Zhang, 2025), RLVR-based training is often viewed as more capable of eliciting advanced reasoning behaviors (Zeng et al., 2025; Yu et al., 2025a). This approach is exemplified by DeepSeek-R1-Zero (Guo et al., 2025) and OpenAI o1 (Jaech et al., 2024), which execute complex reasoning processes through reflection and validation. Following the success of RLVR, a significant body of research has investigated the efficacy of RLVR on popular open-source LLMs, including Qwen (Yang et al., 2024), Mistral (Jiang et al., 2024), and LLaMA (Grattafiori et al., 2024).

This has fostered an optimistic view that RLVR can not only enhance existing model capabilities but also enable the acquisition of novel reasoning knowledge, facilitating a path toward continuous self-improvement (Zeng et al., 2025; Yu et al., 2025a; Wen et al., 2026). RLVR training has been shown to grant LLMs controllable output length for efficient inference (Yan et al., 2025; Cheng et al., 2025; Li et al., 2025c), deepen their reasoning pathways (Bensal et al., 2025), mitigate their weaknesses (Liang et al., 2025, 2026), enable the use of external tools (Rainone et al., 2025; Jin et al., 2025), and even facilitate unsupervised reasoning (Zuo et al., 2025). However, some studies (Yue et al., 2025) argue that despite improving reasoning confidence and reliability, RLVR may inadvertently suppress exploration. By maximizing expected rewards, RLVR tends to reinforce high-reward trajectories while underweighting novel but uncertain ones. This tension between exploration and exploitation remains a core challenge in RLVR.

C.3 Exploration and Exploitation in RLVR for LLMs

Recent studies on exploration and exploitation in RLVR for LLMs have largely relied on token-level analyses of the predictive distribution (Wang et al., 2025; Cui et al., 2025). In this line of work, next-token entropy is commonly treated as a proxy for exploration, motivating methods such as entropy regularization to encourage diverse reasoning paths (Deng et al., 2025a; Cheng et al., 2026). Conversely, lower entropy or higher token-level confidence is often viewed as a sign of exploitation (Fu et al., 2025). Building on this view, several approaches use confidence-related signals to assess, filter, or reinforce reasoning trajectories (Damani et al., 2026; Qiao et al., 2025; Li et al., 2025b; Xue et al., 2026a). However, these measures remain closely tied to token-level uncertainty, which tends to entangle exploration and exploitation under the same measurement framework.

In contrast, we move beyond token-level action statistics and study exploration and exploitation in semantic space at the response level, aiming to decouple and improve both during RL training.

C.4 Representation Dynamics in Deep Reinforcement Learning

Beyond RLVR for LLMs, classical deep RL has long studied how neural representations evolve during training and how they affect exploration, sample efficiency, and generalization. State representation learning for control aims to learn compact latent states that preserve task-relevant dynamics and improve the stability and efficiency of downstream RL algorithms (Lesort et al., 2018). More recent work directly analyzes representation dynamics in deep RL, showing that feature collapse or capacity loss can emerge during training and harm performance (Kumar et al., 2021; Lyle et al., 2022). Other studies further examine broader representational properties, such as dynamics-awareness and orthogonality, and relate them to transfer performance across agents and tasks (Wang et al., 2024a). These findings suggest that representation geometry is closely tied to RL performance, motivating our focus on semantic-space dynamics in RLVR.

D Details of Theorems

D.1 Proof of Theorem 3.1

Suppose we have a matrix of embeddings $\mathbf{Z} \in \mathbb{R}^{T\times D}$*, which is nonzero. Then the effective rank of* $\mathbf{Z}$ *is a lower bound of* $\text{rank}(\mathbf{Z})$*:*

$$1 \leqslant \text{erank}(\mathbf{Z}) \leqslant \text{rank}(\mathbf{Z}) \leqslant \min\{T, D\} \quad (14)$$

Proof. Let the singular value distribution of the matrix $\mathbf{Z}$ be $\mathbf{p} = (p_1, p_2, \ldots, p_{min\{T,D\}})$. The Shannon entropy of this distribution $H(\mathbf{p})$ is bounded. Its minimum is 0, which occurs when only one element of p is 1 and all others are 0. Its maximum is $\log k$, where k is the number of non-zero singular values, and this occurs when the distribution is uniform ($p_j = 1/k$ for all non-zero values). The lower bound is established from the minimum entropy value:

$$\text{erank}(\mathbf{Z}) = \exp(H(\mathbf{p})) \geqslant \exp(0) = 1 \quad (15)$$

Equality holds if and only if the singular value distribution is $(1, 0, \ldots, 0)$, meaning $\mathbf{Z}$ has only one non-zero singular value. For the upper bound, let $k = \text{rank}(\mathbf{Z})$ be the number of non-zero singular values of $\mathbf{Z}$. The entropy of the distribution p is calculated only over these k values and is maximized when they are uniform. Therefore

$$H(\mathbf{p}) \leqslant \log k \quad (16)$$

Applying the exponential function to this inequality gives:

$$\begin{aligned} \text{erank}(\mathbf{Z}) &= \exp\big(H(\mathbf{p})\big) \\ &\leqslant \exp(\log k) = k = \text{rank}(\mathbf{Z})\,. \end{aligned} \quad (17)$$

This establishes that the effective rank is upper-bounded by the conventional rank. The final inequality, $\text{rank}(\mathbf{Z}) \leqslant \min\{T, D\}$, is a standard property of matrix rank. Equality for $\text{erank}(\mathbf{Z}) = \text{rank}(\mathbf{Z})$ holds if and only if the non-zero singular values are all equal, corresponding to a uniform singular value distribution over its support.

D.2 Proof of Proposition 3.5

Assume a hidden-state matrix has effective support on k *approximately orthogonal directions. Then ER and its corresponding ERV are upper-bounded linearly by* k*: ER* $= \mathcal{O}(k)$ *and* $\Delta^{(1)}_{ER} = \mathcal{O}(k)$*. In contrast, the corresponding ERA does not scale with* k*:* $\Delta^{(2)}_{ER} = \mathcal{O}(1)$*.*

Proof. Without loss of generality, we take the effective rank for example. We adopt the provided definition of effective rank for a representation matrix $\mathbf{Z}$ with singular values $\{\sigma_i\}$:

$$\begin{aligned} &\text{erank}(\mathbf{Z}) = \exp\left(-\sum_j p_j \log(p_j)\right), \\ &\text{where} \quad p_j = \frac{\sigma_j}{\sum_k \sigma_k}. \end{aligned} \quad (18)$$

Our analysis focuses on the dataset matrix $\overline{\mathbf{Z}} \in \mathbb{R}^{N\times D}$, whose rows $\{\mathbf{q}_i\}_{i=1}^N$ are the mean token embeddings of N responses. The singular values $\sigma_i(\overline{\mathbf{Z}})$ of $\overline{\mathbf{Z}}$ are the square roots of the eigenvalues of the Gram matrix $\overline{\mathbf{K}} = \overline{\mathbf{Z}}\overline{\mathbf{Z}}^\top$; i.e., $\sigma_j(\overline{\mathbf{Z}}) = \sqrt{\lambda_j(\overline{\mathbf{K}})}$. Given that the rows of $\overline{\mathbf{Z}}$ are nearly orthogonal, the Gram matrix $\overline{\mathbf{K}}$ is strongly diagonal-dominant. Its eigenvalues can be approximated by its diagonal entries, for $j = 1, \ldots, N$:

$$\lambda_j(\overline{\mathbf{K}}) \approx \overline{\mathbf{K}}_{jj} = \|\mathbf{q}_j\|^2 = \frac{1}{T} \quad (19)$$

The matrix has N significant eigenvalues, each approximately equal to $1/T$. The singular values of $\overline{\mathbf{Z}}$ are the square roots of the eigenvalues of $\overline{\mathbf{K}}$, for $j = 1, \ldots, N$:

$$\sigma_j(\overline{\mathbf{Z}}) = \sqrt{\lambda_j(\overline{\mathbf{K}})} \approx \sqrt{\frac{1}{T}} = \frac{1}{\sqrt{T}} \tag{20}$$

To calculate the effective rank, we first normalize these singular values to form a probability distribution $\{p_j\}$. The sum of singular values is:

$$\sum_{k=1}^{N} \sigma_k(\overline{\mathbf{Z}}) \approx \sum_{k=1}^{N} \frac{1}{\sqrt{T}} = \frac{N}{\sqrt{T}} \tag{21}$$

The individual probabilities are therefore:

$$p_j = \frac{\sigma_j}{\sum_k \sigma_k} \approx \frac{1/\sqrt{T}}{N/\sqrt{T}} = \frac{1}{N} \tag{22}$$

The distribution $\mathbf{p} = \{p_1, p_2, \ldots, p_N\}$ is a uniform distribution over N states. The Shannon entropy of this distribution is maximal:

$$\begin{aligned} H(\mathbf{p}) &= -\sum_{j=1}^{N} p_j \log(p_j) = -\sum_{j=1}^{N} \frac{1}{N} \log\left(\frac{1}{N}\right) \\ &= -N\left(\frac{-\log(N)}{N}\right) = \log(N)\,. \end{aligned} \tag{23}$$

The effective rank is the exponential of this entropy: $\text{erank}(\overline{\mathbf{Z}}) = \exp(H(\mathbf{p})) = \exp(\log(N)) = N$. In the maximal prompt entropy regime, the effective rank of the dataset matrix scales as $\mathcal{O}(N)$.

We adapt them to our context by defining the metric's value at "time" n as the Effective Rank computed on the dataset matrix formed by the first n prompts, denoted $\overline{\mathbf{Z}}_{1:n}$. Let $m_n = \text{erank}(\overline{\mathbf{Z}}_{1:n})$. From our previous analysis, we established a crucial result that forms the basis of this derivation: for maximal cases, the effective rank of a dataset with n prompts scales linearly with n.

$$m_n = \text{erank}(\overline{\mathbf{Z}}_{1:n}) \approx n \tag{24}$$

We will use this linear approximation to derive the scaling orders of the difference metrics, assuming a stride of $s = 1$ for simplicity. The first-order difference quantifies the average "velocity" of change in the metric relative to its historical mean. Instantaneous First-Order Difference ($\delta_n^{(1)}$) is the value at step n minus the average of all preceding values.

$$\delta_n^{(1)} = m_n - \left(\frac{1}{n-1}\sum_{k=1}^{n-1} m_k\right) \tag{25}$$

Substituting our approximation $m_k \approx k$:

$$\delta_n^{(1)} \approx n - \left(\frac{1}{n-1}\sum_{k=1}^{n-1} k\right) \tag{26}$$

$$\delta_n^{(1)} \approx n - \left(\frac{1}{n-1} \cdot \frac{(n-1)n}{2}\right) = n - \frac{n}{2} = \frac{n}{2} \tag{27}$$

The historical-baseline deviation grows linearly with n. Overall First-Order Difference $\Delta_{\text{erank}}^{(1)}$: This is the average of the historical-baseline deviation over the entire dataset of size N.

$$\Delta_{\text{erank}}^{(1)} = \frac{1}{N-1}\sum_{n=2}^{N} \delta_n^{(1)} \approx \frac{1}{N-1}\sum_{n=2}^{N} \frac{n}{2} \tag{28}$$

$$\begin{aligned} \Delta_{\text{erank}}^{(1)} &\approx \frac{1}{2(N-1)}\left(\left(\sum_{n=1}^{N} n\right) - 1\right) \\ &= \frac{1}{2(N-1)}\left(\frac{N(N+1)}{2} - 1\right). \end{aligned} \tag{29}$$

For large N, the expression is dominated by the highest power of N:

$$\Delta_{\text{erank}}^{(1)} \sim \frac{N^2/4}{N} = \frac{N}{4} \tag{30}$$

The first-order difference of the effective rank scales linearly with the number of prompts, $\Delta_{\text{erank}}^{(1)} = \mathcal{O}(N)$. As for second-order difference, we compute the change in historical-baseline deviation between consecutive values of $\delta_n^{(1)}$.

$$\delta_n^{(1)} - \delta_{n-1}^{(1)} \approx \frac{n}{2} - \frac{n-1}{2} = \frac{1}{2} \tag{31}$$

This change is a constant, indicating a linear increase in the first-order difference. Overall Second-Order Difference $\Delta_{\text{erank}}^{(2)}$:

$$\begin{aligned} \Delta_{\text{erank}}^{(2)} &= \frac{1}{N-2}\sum_{n=3}^{N} \left(d_n^{(1)} - d_{n-1}^{(1)}\right) \\ &\approx \frac{1}{N-2}\sum_{n=3}^{N} \frac{1}{2}\,. \end{aligned} \tag{32}$$

$$\Delta_{\text{erank}}^{(2)} \approx \frac{1}{N-2} \cdot (N-2) \cdot \frac{1}{2} = \frac{1}{2} \tag{33}$$

The second-order difference of the effective rank is constant and does not depend on N, yielding a scaling order of $\Delta_{\text{erank}}^{(2)} = \mathcal{O}(1)$.

E Additional Theoretical Support for Exploration and Exploitation Metrics

In this section we formalize the relationship between our proposed hidden-state metrics (*Effective Rank* and *Effective Rank Velocity*) and the classical notions of exploration and exploitation in reinforcement learning. We first show that the token-level metrics in action space (average log probability and response entropy) are algebraically coupled, whereas the hidden-state metrics in semantic space are not. We then provide a representation-level justification for interpreting Effective Rank as a measure of semantic exploration, and Effective Rank Velocity as a measure of representation-level exploitation that is strongly correlated with greedy value improvement under the PPO-style architecture used in RLVR. Throughout, we consider a conditional language model $p_\theta(y \mid x)$ and a Transformer backbone that produces hidden states $z_t \in \mathbb{R}^D$ at each time step t for a given prompt x and generated response $y_{1:T}$.

E.1 Action vs. Semantic Exploration–Exploitation Metrics

In this subsection, we formalize the difference between the token-level *action* metrics used in prior RLHF/RLVR work and the hidden-state *semantic* metrics proposed in this paper. For a given prompt x and generated response $y_{1:T}$, let $\pi_\theta(\cdot \mid x, y_{<t})$ denote the model's token-level policy distribution at step t, i.e. the softmax over the vocabulary induced by the logits at that position.

Token-level action space, log-probability, and entropy. We define the *average log probability* of a response and the *response entropy* as

$$\begin{aligned} \mathrm{AvgLogProb}(x, y_{1:T}) &:= \frac{1}{T}\sum_{t=1}^{T} \log \pi_\theta\big(y_t \mid x, y_{<t}\big), \\ \mathrm{RespEnt}(x, y_{1:T}) &:= \frac{1}{T}\sum_{t=1}^{T} H(\pi_\theta(\cdot \mid x, y_{<t})), \\ H(p) &:= -\sum_v p(v)\log p(v), \\ p(v) &:= \pi_\theta(v \mid x, y_{<t}). \end{aligned} \tag{34}$$

Thus $\mathrm{RespEnt}$ is the *token-level* entropy averaged over the response: at each step we compute the Shannon entropy of the vocabulary distribution and then average over time. In the response-level semantic space, we consider x drawn from a prompt distribution $\mathcal{P}_x$ and, for the purpose of analysis, responses $y_{1:T}$ drawn *on-policy* from the model $p_\theta(\cdot \mid x)$:

$$\begin{aligned} &\mathcal{L}_{\text{avg-log}}(\theta) := \\ &\mathbb{E}_{x\sim p(x)}\, \mathbb{E}_{y_{1:T}\sim p_\theta(\cdot|x)}\Big[\mathrm{AvgLogProb}(x, y_{1:T})\Big], \\ &\mathcal{H}_{\text{resp}}(\theta) := \\ &\mathbb{E}_{x\sim p(x)}\, \mathbb{E}_{y_{1:T}\sim p_\theta(\cdot|x)}\Big[\mathrm{RespEnt}(x, y_{1:T})\Big]. \end{aligned} \tag{35}$$

Proposition E.1 (Token-level exploitation and exploration are tightly coupled)**.** *Under on-policy sampling $y_{1:T} \sim p_\theta(\cdot \mid x)$, the corpus-level average log probability $\mathcal{L}_{avg\text{-}log}(\theta)$ and response entropy $\mathcal{H}_{resp}(\theta)$ satisfy*

$$\mathcal{L}_{avg\text{-}log}(\theta) \;=\; -\mathcal{H}_{resp}(\theta). \tag{36}$$

In particular, under the same sampling distribution, any change of the model that increases token-level exploitation in action space as measured by $\mathcal{L}_{avg\text{-}log}$ necessarily decreases $\mathcal{H}_{resp}$ by the same amount (and vice versa).

Proof. Fix a prompt x and a time step t. Conditioned on x and the history $y_{<t}$, the next token y_t is drawn from $\pi_\theta(\cdot \mid x, y_{<t})$. Taking the expectation of $\log \pi_\theta(y_t \mid x, y_{<t})$ under this distribution yields:

$$\begin{aligned} &\mathbb{E}_{y_t\sim\pi_\theta(\cdot|x,y_{<t})}\Big[\log \pi_\theta(y_t \mid x, y_{<t})\Big] \\ &= \sum_v \pi_\theta(v \mid x, y_{<t}) \log \pi_\theta(v \mid x, y_{<t}) \\ &= -H(\pi_\theta(\cdot \mid x, y_{<t})). \end{aligned} \tag{37}$$

Now consider a full response $y_{1:T} \sim p_\theta(\cdot \mid x)$. By the law of iterated expectations,

$$\begin{aligned} &\mathbb{E}_{y_{1:T}\sim p_\theta(\cdot|x)}\Big[\log \pi_\theta(y_t \mid x, y_{<t})\Big] \\ &= \mathbb{E}_{y_{<t}\sim p_\theta(\cdot|x)}\Big[\mathbb{E}_{y_t\sim\pi_\theta(\cdot|x,y_{<t})}\big[\log \pi_\theta(y_t \mid x, y_{<t})\big]\Big] \\ &= -\mathbb{E}_{y_{<t}\sim p_\theta(\cdot|x)}\Big[H(\pi_\theta(\cdot \mid x, y_{<t}))\Big]. \end{aligned} \tag{38}$$

where we used Eq. 37 in the last step. Averaging over $t = 1, \dots, T$ and dividing by T gives

$$\begin{aligned}
&\mathbb{E}_{y_{1:T} \sim p_\theta(\cdot|x)}\Big[\text{AvgLogProb}(x, y_{1:T})\Big] \\
&= \mathbb{E}_{y_{1:T} \sim p_\theta(\cdot|x)}\Big[\frac{1}{T}\sum_{t=1}^{T} \log \pi_\theta(y_t \mid x, y_{<t})\Big] \\
&= -\mathbb{E}_{y_{1:T} \sim p_\theta(\cdot|x)}\Big[\frac{1}{T}\sum_{t=1}^{T} H(\pi_\theta(\cdot \mid x, y_{<t}))\Big] \\
&= -\mathbb{E}_{y_{1:T} \sim p_\theta(\cdot|x)}\Big[\text{RespEnt}(x, y_{1:T})\Big]. \qquad (39)
\end{aligned}$$

Finally, taking expectation over prompts $x \sim p(x)$ on both sides of Eq. 39 yields

$$\mathcal{L}_{\text{avg-log}}(\theta) = -\mathcal{H}_{\text{resp}}(\theta), \qquad (40)$$

which is exactly Eq. 36. This shows that under on-policy sampling, the two token-level metrics are related by a fixed negative sign and thus cannot be decoupled in the action space. □

Hidden-state Effective Rank and velocity. The next proposition shows that these two hidden-state metrics in Sec. 3.1 and 3.3 are *structurally decoupled* at the level of trajectories: knowing the final Effective Rank alone does not determine ERV, and conversely.

Proposition E.2 (Hidden-state metrics are structurally decoupled)**.** *Fix $K \geqslant 3$ evaluation steps. Consider the map that associates to each Effective Rank trajectory $m = (m_1, \dots, m_K) \in \mathbb{R}^K$ its final value*

$$\text{ER}_{\textit{final}}(m) := m_K \qquad (41)$$

and its Effective Rank velocity

$$\text{ERV}(m) := \frac{1}{K-1}\sum_{j=2}^{K}\Big(m_j - \frac{1}{j-1}\sum_{k=1}^{j-1} m_k\Big). \qquad (42)$$

Then:

1. *There is no function $f : \mathbb{R} \to \mathbb{R}$ such that $\text{ERV}(m) = f(\text{ER}_{\textit{final}}(m))$ for all trajectories $m \in \mathbb{R}^K$.*

2. *There is no function $g : \mathbb{R} \to \mathbb{R}$ such that $\text{ER}_{\textit{final}}(m) = g(\text{ERV}(m))$ for all trajectories $m \in \mathbb{R}^K$.*

Equivalently, $\text{ER}_{\textit{final}}$ *and* ERV *are not functionally dependent: they capture genuinely different aspects of the Effective Rank sequence.*

Proof. We view ER_{final} and ERV as real-valued functions on $\mathbb{R}^K$. The proof is purely algebraic and does not rely on any monotonicity of m_j.

Step 1: ERV is a non-trivial linear functional. Introduce the shorthand

$$\Delta m_j := m_j - m_{j-1}, \quad j \geqslant 2. \qquad (43)$$

A direct calculation shows that each increment δ_j can be written as

$$\delta_j = \frac{1}{j-1}\sum_{r=2}^{j}(r-1)\,\Delta m_r, \quad j \geqslant 2, \qquad (44)$$

so that ERV is a linear functional of m:

$$\text{ERV}(m) = \sum_{j=1}^{K} \alpha_j m_j, \qquad (45)$$

for some fixed coefficients $\alpha_1, \dots, \alpha_K$ that depend only on K (and s) and satisfy $\sum_{j=1}^{K} \alpha_j = 0$ and $\alpha_j \neq 0$ for at least two indices j (e.g. $\alpha_1 \neq 0$ and $\alpha_K \neq 0$). In particular, ERV is *not* proportional to the projection onto any single coordinate m_j.

Step 2: No functional dependence of ERV on the final ER. Suppose, for contradiction, that there exists a function $f : \mathbb{R} \to \mathbb{R}$ such that

$$\begin{aligned}
\text{ERV}(m) &= f(\text{ER}_{\text{final}}(m)) \\
&= f(m_K) \quad \text{for all } m \in \mathbb{R}^K.
\end{aligned} \qquad (46)$$

Fix any constant $c \in \mathbb{R}$. Consider the affine subspace

$$\mathcal{A}_c := \{m \in \mathbb{R}^K : m_K = c\}. \qquad (47)$$

On this subspace, $\text{ER}_{\text{final}}(m) \equiv c$ is constant, so by assumption $\text{ERV}(m) \equiv f(c)$ must also be constant. However, ERV is a non-trivial linear functional that depends on at least one coordinate m_j with $j < K$. Therefore, restricted to $\mathcal{A}_c$, the map $m \mapsto \text{ERV}(m)$ varies with those coordinates and cannot be constant. This yields a contradiction. Hence no such f exists.

Step 3: No functional dependence of final ER on ERV. The argument is symmetric. Suppose there exists $g : \mathbb{R} \to \mathbb{R}$ such that

$$\begin{aligned}
m_K &= \text{ER}_{\text{final}}(m) \\
&= g(\text{ERV}(m)) \quad \text{for all } m \in \mathbb{R}^K.
\end{aligned} \qquad (48)$$

Fix any constant $c \in \mathbb{R}$ and consider the affine subspace

$$\mathcal{B}_c := \{m \in \mathbb{R}^K : \text{ERV}(m) = c\}. \qquad (49)$$

Since ERV is a non-trivial linear functional, $\mathcal{B}_c$ is an affine hyperplane of codimension 1, and m_K can vary freely among its points. Yet the assumed relation $m_K = g(\mathrm{ERV}(m)) = g(c)$ would force m_K to be constant on $\mathcal{B}_c$, which is impossible. Thus no such g exists.

Combining the two steps, we conclude that $\mathrm{ER}_{\text{final}}$ and ERV are not functionally dependent on each other. □

Summary. Proposition E.1 establishes that two classical token-level metrics, average log probability and response entropy, are *algebraically coupled* under on-policy sampling and therefore cannot be varied independently. In contrast, Proposition E.2 shows that our hidden-state metrics, terminal Effective Rank and Effective Rank Velocity, are structurally decoupled. They depend on different functionals of the Effective Rank trajectory and capture complementary aspects of exploration (semantic diversity level) and exploitation (the rate of semantic diversity gain).

E.2 Effective Rank as Semantic Exploration

We now formalize the interpretation of Effective Rank as a measure of semantic diversity and uncertainty in the hidden-state space, and hence as a representation-level proxy for exploration in LLM reasoning. We assume that the hidden states are semantic representations in the sense that downstream semantic properties can be approximately recovered as linear functionals of the hidden vectors. This is standard in representation learning and probing work on large language models.

Assumption E.3 (Semantic linear decodability)**.** There exists a collection of K semantic features $s^{(1)}, \ldots, s^{(K)}$ (e.g., semantic roles, entity identities, factual attributes, intermediate reasoning states) such that for each time step t and feature index k we have

$$s_t^{(k)} \;\approx\; w_k^\top h_t, \quad w_k \in \mathbb{R}^D. \tag{50}$$

Semantic features are therefore approximately linearly decodable from hidden states.

Assumption E.4 (Bounded energy and finite support)**.** For a given trajectory $Z_{1:T}$, there exists an orthonormal basis of semantic directions $\{e_1, \ldots, e_D\}$ such that each hidden state admits a decomposition

$$z_t = \sum_{i=1}^{D} a_{t,i}\, e_i, \tag{51}$$

with $\sum_{t=1}^{T} a_{t,i}^2 < \infty$ for all i, and only finitely many coordinates $a_{t,i}$ carry task-relevant semantic variation.

Proposition E.5 (Effective Rank as semantic diversity and uncertainty)**.** *Let $Z_{1:T}$ have singular values $\{\sigma_i\}_{i=1}^{\min(T,D)}$ (i.e., the diagonal of Σ in its SVD), and let $\mathrm{ER}(Z_{1:T})$ denote its Effective Rank.*

1. *If the trajectory uses exactly k orthogonal semantic directions with equal energy and no others, i.e. the singular values satisfy $\sigma_1 = \cdots = \sigma_k > 0$ and $\sigma_{k+1} = \cdots = \sigma_r = 0$, then $\mathrm{ER}(Z_{1:T}) = k$.*

2. *More generally, if the singular value spectrum of $Z_{1:T}$ becomes more* spread out *over more directions in the sense of majorization (i.e. the normalized singular value vector becomes more uniform over a larger support), then $\mathrm{ER}(Z_{1:T})$ increases.*

Consequently, $\mathrm{ER}(Z_{1:T})$ is a basis-invariant, strictly increasing measure of the number of independent semantic directions that are effectively used by the hidden states, and thus a natural representation-level proxy for semantic exploration and uncertainty.

Proof. We proceed in two parts.

(1) Equal-energy k-dimensional semantic subspace. Suppose $Z_{1:T}$ uses exactly k orthogonal semantic directions with equal energy. Then, up to permutation, the non-zero singular values satisfy

$$\sigma_1 = \cdots = \sigma_k = c > 0, \qquad \sigma_{k+1} = \cdots = \sigma_r = 0. \tag{52}$$

The normalized singular values are thus

$$q_i = \begin{cases} 1/k, & i = 1, \ldots, k, \\ 0, & i > k, \end{cases} \tag{53}$$

and the entropy of q is

$$H(q) = -\sum_{i=1}^{k} \frac{1}{k} \log \frac{1}{k} = \log k. \tag{54}$$

Therefore

$$\mathrm{ER}(Z_{1:T}) = \exp(H(q)) = \exp(\log k) = k. \tag{55}$$

This shows that, in the idealized case of exactly k equi-energic semantic directions, Effective Rank matches the true semantic dimensionality k.

(2) Monotonicity under majorization. Consider two hidden-state trajectories Z and $\tilde{Z}$ with normalized singular value spectra q and $\tilde{q}$, respectively. Suppose that q is *majorized* by $\tilde{q}$ (denoted $q \prec \tilde{q}$), meaning intuitively that $\tilde{q}$ is "more spread out" and therefore more uniform across a larger support.

It is a standard result in information theory that the Shannon entropy $H(\cdot)$ is Schur-concave: if $q \prec \tilde{q}$, then $H(q) \leqslant H(\tilde{q})$ with strict inequality whenever $q \neq \tilde{q}$. Therefore

$$\mathrm{ER}(Z) = \exp(H(q)) \;\leqslant\; \exp(H(\tilde{q})) = \mathrm{ER}(\tilde{Z}), \tag{56}$$

with strict inequality when the majorization is strict. In words, whenever the singular value spectrum becomes more spread out across more directions, the Effective Rank strictly increases.

Summary. Combining the two parts, we see that Effective Rank equals the number of equi-energic semantic directions in the idealized case and increases whenever the representation distributes energy over more orthogonal directions. Since, by Assumption E.3, semantic features are linearly decodable along such directions, $\mathrm{ER}(Z_{1:T})$ provides a basis-invariant measure of how many independent semantic dimensions are explored by the hidden states and how evenly they are used. This justifies its interpretation as a representation-level exploration and uncertainty metric. □

E.3 Effective Rank Velocity as Semantic Exploitation

Building on Sec. 3.1, where Effective Rank (ER) is shown to measure the number and uniform use of semantic directions in the hidden-state space, we now give a representation-only justification for interpreting Effective Rank Velocity (ERV) as *semantic exploitation*.

Throughout this subsection we fix a single trajectory $Z_{1:T}$ and a stride s. Let the evaluation positions be $t_j = js$ for $j = 1, \dots, K$ with $K = \lfloor (T-1)/s \rfloor$, and write

$$m_j \;:=\; \mathrm{ER}\big(Z_{1:t_j}\big), \qquad j = 1, \dots, K. \tag{57}$$

Thus, $\{m_j\}_{j=1}^{K}$ is the ER trajectory of the growing prefixes of the same response.

ERV as a recency-weighted sum of ER increments. For convenience we recall the notation of Def. 3.3 with $M = \mathrm{ER}$. Define the local ER increments

$$\Delta m_r \;:=\; m_r - m_{r-1}, \quad r \geqslant 2. \tag{58}$$

Def. 3.3 introduces the "historical-baseline deviation"

$$\delta_j \;:=\; m_j - \frac{1}{j-1} \sum_{k=1}^{j-1} m_k, \qquad j \geqslant 2, \tag{59}$$

and the first-order temporal difference (ERV) as

$$\Delta_{\mathrm{ER}}^{(1)} \;:=\; \frac{1}{K-1} \sum_{j=2}^{K} \delta_j. \tag{60}$$

The following lemma makes explicit that ERV is a recency-weighted average of the consecutive ER increments.

Lemma E.6 (Recency-weighted velocity of ER)**.** *For any sequence $(m_j)_{j=1}^{K}$, the historical-baseline deviation admit the representation*

$$\delta_j = \frac{1}{j-1} \sum_{r=2}^{j} (r-1)\, \Delta m_r, \qquad j \geqslant 2, \tag{61}$$

and hence ERV can be written as

$$\begin{aligned} \Delta_{ER}^{(1)} &= \sum_{r=2}^{K} w_r\, \Delta m_r, \\ w_r &:= \frac{r-1}{K-1} \sum_{j=r}^{K} \frac{1}{j-1} > 0. \end{aligned} \tag{62}$$

In particular, ERV is a positive linear combination of the local ER increments Δm_r, assigning larger weights to more recent steps.

Proof. Eq. 61 is exactly in Def. 3.3 with $M = \mathrm{ER}$, obtained by expressing δ_j in terms of the increments Δm_r via telescoping. Plugging Eq. 61 into the definition of $\Delta_{\mathrm{ER}}^{(1)}$ and exchanging the order of summation yields

$$\begin{aligned} \Delta_{\mathrm{ER}}^{(1)} &= \frac{1}{K-1} \sum_{j=2}^{K} \frac{1}{j-1} \sum_{r=2}^{j} (r-1)\, \Delta m_r \\ &= \sum_{r=2}^{K} \left[\frac{r-1}{K-1} \sum_{j=r}^{K} \frac{1}{j-1} \right] \Delta m_r. \end{aligned} \tag{63}$$

which gives Eq. 62 with w_r as stated. Since $r-1 > 0$ and the harmonic tail $\sum_{j=r}^{K} (j-1)^{-1}$ is positive, we have $w_r > 0$ for all r. □

Semantic exploitation as positive ER drift in a fixed semantic subspace. Sec. E.1 states that, up to an orthonormal change of basis, the hidden states can be written as $h_t = \sum_i a_{t,i} e_i$ with bounded energy along each semantic direction e_i, and that ER is a strictly increasing, basis-invariant measure of how many semantic directions are effectively used and how evenly energy is distributed among them. In particular, if the set of active directions (support of the singular value spectrum) is kept fixed and the spectrum becomes more uniform (in the sense of majorization), then ER strictly increases. Motivated by this, we isolate an idealized *semantic exploitation* regime in which the trajectory has already selected a semantic subspace and is refining it.

Definition E.7 (Semantic exploitation regime)**.** Let $(m_j)_{j=1}^K$ be the ER trajectory of a response, and let $q^{(j)}$ denote the normalized singular value vector of $Z_{1:t_j}$. We say that steps $j = 2, \dots, K$ form a *semantic exploitation regime with rate* $\mu > 0$ if:

1. (Fixed semantic support) The support of $q^{(j)}$ is independent of j, i.e., the set of active semantic directions is fixed.

2. (Uniformization within the support) For every $j \geqslant 2$, $q^{(j)}$ is more uniform than $q^{(j-1)}$ on this fixed support, in the sense of majorization, so that by Prop. E.5 we have $m_j - m_{j-1} = \Delta m_j \geqslant \mu$ for some $\mu > 0$.

Intuitively, condition (i) says that the model has committed to a particular semantic subspace (a line of reasoning), and condition (ii) says that it keeps redistributing energy within this subspace to make use of all its semantic directions more evenly. This is precisely the notion of "refining a promising strategy" in representation space.

ERV lower-bounds the semantic exploitation rate. Under Def. E.7, ER experiences a persistent positive drift along the trajectory. The next proposition shows that ERV is a quantitative lower bound on this drift, and thus a natural measure of semantic exploitation strength.

Proposition E.8 (ERV as a lower bound on semantic exploitation rate)**.** *Assume the hidden states satisfy Assumptions E.3 and E.4 and that steps $j = 2, \dots, K$ form a semantic exploitation regime with rate $\mu > 0$ in the sense of Def. E.7, so that $\Delta m_j \geqslant \mu$ for all $j \geqslant 2$. Then*

$$\Delta_{ER}^{(1)} \geqslant \frac{\mu K}{4}. \tag{64}$$

In particular, ERV is strictly positive and grows linearly with the length K of the exploitation segment.

Proof. By Eq. 61 and the assumption $\Delta m_r \geqslant \mu$ we obtain, for each $j \geqslant 2$,

$$\begin{aligned} \delta_j &= \frac{1}{j-1} \sum_{r=2}^{j} (r-1)\, \Delta m_r \\ &\geqslant \frac{1}{j-1} \sum_{r=2}^{j} (r-1)\, \mu \\ &= \frac{\mu}{j-1} \sum_{r=2}^{j} (r-1) = \frac{\mu j}{2}. \end{aligned} \tag{65}$$

Averaging over j then yields

$$\begin{aligned} \Delta_{\mathrm{ER}}^{(1)} &= \frac{1}{K-1} \sum_{j=2}^{K} \delta_j \\ &\geqslant \frac{1}{K-1} \sum_{j=2}^{K} \frac{\mu j}{2} = \frac{\mu}{2(K-1)} \sum_{j=2}^{K} j. \end{aligned} \tag{66}$$

For $K \geqslant 2$ we haven $\sum_{j=2}^{K} j \geqslant \frac{K(K-1)}{2}$, so

$$\Delta_{\mathrm{ER}}^{(1)} \geqslant \frac{\mu}{2(K-1)} \cdot \frac{K(K-1)}{2} = \frac{\mu K}{4}, \tag{67}$$

which proves Eq. 64. □

Thus, in an idealized regime where the model has already discovered a useful semantic subspace and is consistently enriching it, ERV provides a strictly positive, linearly growing lower bound on the rate at which semantic complexity within that subspace is being exploited.

F Implementation Details

F.1 Training and Evaluation Details

We typically use the same set of hyperparameters to train and evaluate all models in the SimpleRL-Zoo series (Zeng et al., 2025) in the default main experiment setting.

Training. We conduct all experiments with 4 A800-PCIE-80GB GPUs. For GRPO and PPO, we use a prompt batch size of 48 with a maximum rollout length of 1536 tokens. Training is performed using a mini-batch size of 24. For GRPO, we generate 4 rollouts per prompt. For PPO, we use DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025) as the value model and generate 1 rollout per prompt. The default sampling temperature is set to 1.0, and the clip ratio is 0.2. For all actor models ranging from 3B to 8B parameters, we use a learning rate of 1e-6 and a KL loss coefficient of 1e-4. For critic models in PPO, we use a learning rate of 1e-5. For our training datasets, we follow the same setup as in Zeng et al. (2025), where the data is filtered from GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021b) configured with different difficulty levels for models of varying capabilities. We tested using the checkpoint model trained up to 120 steps.

Evaluation. We build our evaluation script based on that of Zeng et al. (2025), using a temperature of 0.6 and a maximum generation length of 2048 tokens. To ensure consistency, we adopt the same prompt template used during training. For most benchmarks, we report Pass@1 results. However, for benchmarks like AIME 2024, which contains fewer problems, we report both Pass@1 and average accuracy (Pass@256), computed over 256 generated samples per problem.

Base Models. To demonstrate the universality of our insights and methods, we conduct zero RL training experiments on Llama-3.2 (3B), Llama-3.1 (8B) (Grattafiori et al., 2024), Mistral-v0.3-7B (Jiang et al., 2024), and Qwen-2.5 (1.5B, 3B, 7B) (Hui et al., 2024). For value model in PPO, we use DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025) for all experiments.

Benchmark. We evaluate on a diverse suite of mathematical reasoning benchmarks. These include standard benchmarks such as GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021b), ASDiv (Miao et al., 2020), Carp (English Version) (Zhang et al., 2023), MAWPS (Koncel-Kedziorski et al., 2016), SVAMP (Patel et al., 2021), TabMWP (Lu et al., 2023), and OlympiadBench (He et al., 2024); Chinese mathematics collections like CMATH (Wei et al., 2023) and Gaokao 2024; and benchmarks from mathematics competitions, including the 2024/2025 AIME and the 2023/2024 AMC.

F.2 Efficient Incremental Computation of Higher-Order Metrics

A naive computation of the temporal difference metrics would be computationally prohibitive. Our method's feasibility hinges on an efficient, incremental algorithm that computes the required metrics without redundant operations on the growing hidden state matrix $\mathbf{Z} \in \mathbb{R}^{T \times D}$.

The effective rank is derived from the singular values of the mean-centered hidden state matrix $\mathbf{Z}_c$. These are equivalent to the square roots of the eigenvalues of the centered Gram matrix $\mathbf{K} = \mathbf{Z}_c\mathbf{Z}_c^\top$. Instead of recomputing $\mathbf{K}_t$ from scratch at each time step t, our algorithm incrementally updates two sufficient statistics: the uncentered Gram matrix $\mathbf{U}_t = \mathbf{Z}_{1:t}\mathbf{Z}_{1:t}^\top$ and the sum of hidden state vectors $s_t = \sum_{i=1}^{t} z_i$. When extending the analysis window, the new uncentered Gram matrix $\mathbf{U}_t$ is constructed from the prior matrix $\mathbf{U}_{t-s}$ and the new chunk of hidden states $\Delta\mathbf{Z}_t = \mathbf{Z}_{t-s+1:t}$. This update follows a recursive block matrix structure:

$$\mathbf{U}_t = \begin{pmatrix} \mathbf{U}_{t-s} & \mathbf{Z}_{1:t-s}(\Delta\mathbf{Z}_t)^\top \\ (\Delta\mathbf{Z}_t)\mathbf{Z}_{1:t-s}^\top & (\Delta\mathbf{Z}_t)(\Delta\mathbf{Z}_t)^\top \end{pmatrix} \tag{68}$$

From the efficiently updated $\mathbf{U}_t$ and s_t, we can directly construct the centered Gram matrix $\mathbf{K}_t$. Letting $\boldsymbol{\mu}_t = s_t/t$ be the mean vector and $\mathbf{1}_t$ be a column vector of ones, $\mathbf{K}_t$ can be expressed as:

$$\begin{aligned}\mathbf{K}_t = \mathbf{U}_t - (\mathbf{Z}_{1:t}\boldsymbol{\mu}_t)\mathbf{1}_t^\top - \mathbf{1}_t(\mathbf{Z}_{1:t}\boldsymbol{\mu}_t)^\top \\ + (\boldsymbol{\mu}_t^\top\boldsymbol{\mu}_t)\cdot(\mathbf{1}_t\mathbf{1}_t^\top).\end{aligned} \tag{69}$$

This allows for the calculation of $\mathbf{K}_t$ without re-accessing the full history of hidden states. The eigenvalues $\{\lambda_j\}$ of $\mathbf{K}_t$ are then used to derive the effective rank. First, the singular values of the centered matrix are obtained, $\sigma_j = \sqrt{\lambda_j}$. These are normalized to form a probability distribution, $p_j = \sigma_j / \sum_k \sigma_k$. The effective rank is then the exponential of the Shannon entropy of this distribution:$\text{erank}(\mathbf{Z}_{c,t}) = \exp\left(-\sum_j p_j \log p_j\right)$. This pipeline efficiently yields a sequence of effective rank values, $m_{j\cdot s} = \text{erank}(\mathbf{Z}_{c,j\cdot s})$, at each stride s. From this sequence, we compute the instantaneous first-order difference δ, which compares the current value to the running average of

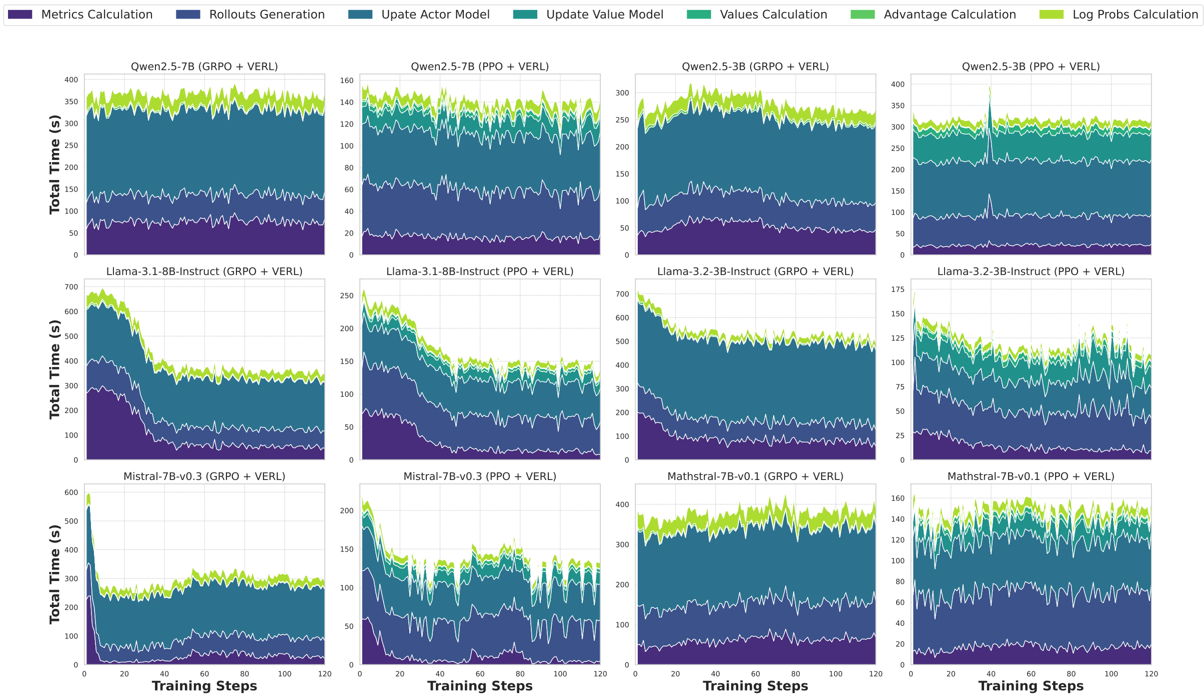


Figure 6: Time overhead of the main computation of RL Training.

all preceding values. This is defined recursively as:$\delta_{j\cdot s} = m_{j\cdot s} - \frac{1}{j-1}\sum_{k=1}^{j-1} m_{k\cdot s}$.

The computational advantage of this incremental approach is substantial. While the total cost for the series of eigenvalue decompositions $\mathcal{O}(T^4/s)$, is common to both methods, the cost of matrix construction differs significantly. The naive method's recalculation totals $\mathcal{O}(DT^3/s)$, whereas our incremental update method reduces this to $O(DT^2)$. This reduction of the polynomial dependency on sequence length T from cubic to quadratic is critical, as this term is multiplied by the large hidden dimension D, making it the dominant factor in practical performance and rendering the dense calculation of temporal dynamics feasible. In the *worst-case* scenario where the sequence length T exceeds the hidden dimension D, and both D and the stride s can be treated as constants. The naïve approach that reconstructs matrices independently at each stride has a matrix-construction cost scaling as $\mathcal{O}(T^2)$, VERL's incremental Gram/covariance updates scale as $\mathcal{O}(T)$. So asymptotically, our implementation has a strictly better dependency on T than a naïve SVD-based design.

F.3 Time Overhead of VERL Training

We conducted post-training with Zero RL on several base models. The Fig. 6 illustrates the time associated with each computational stage. The 'metrics calculation' component, which represents the cost of computing metrics for hidden states, accounts for an insignificant portion of the total processing time. This demonstrates that our method does not introduce substantial time overhead. To further stress-test the *worst-case* scenario, we deliberately compute ER, ERV, and ERA on the CPU rather than the GPU, and still observe that the additional time overhead remains negligible.

G More Experiments

G.1 Detailed Analysis of Response-Level Hidden States

As shown in Figs. 7 and 8, our analysis of response-level hidden states across additional LLMs confirms that the insights presented in Sec. 4.1 hold true for various base models and RL paradigms.

G.2 Detailed Analysis of Dataset-Level Hidden States

As shown in Figs. 9 and 10, our analysis of dataset-level hidden states across additional LLMs confirms that the insights presented in Sec. 4.2 hold for various base models and RL paradigms.

G.3 Detailed Analysis of Pass@1 Performance

As shown in Tab. 4, Pass@1 measures the model's ability to generate a correct answer in a single attempt, which directly reflects its exploitation abil-

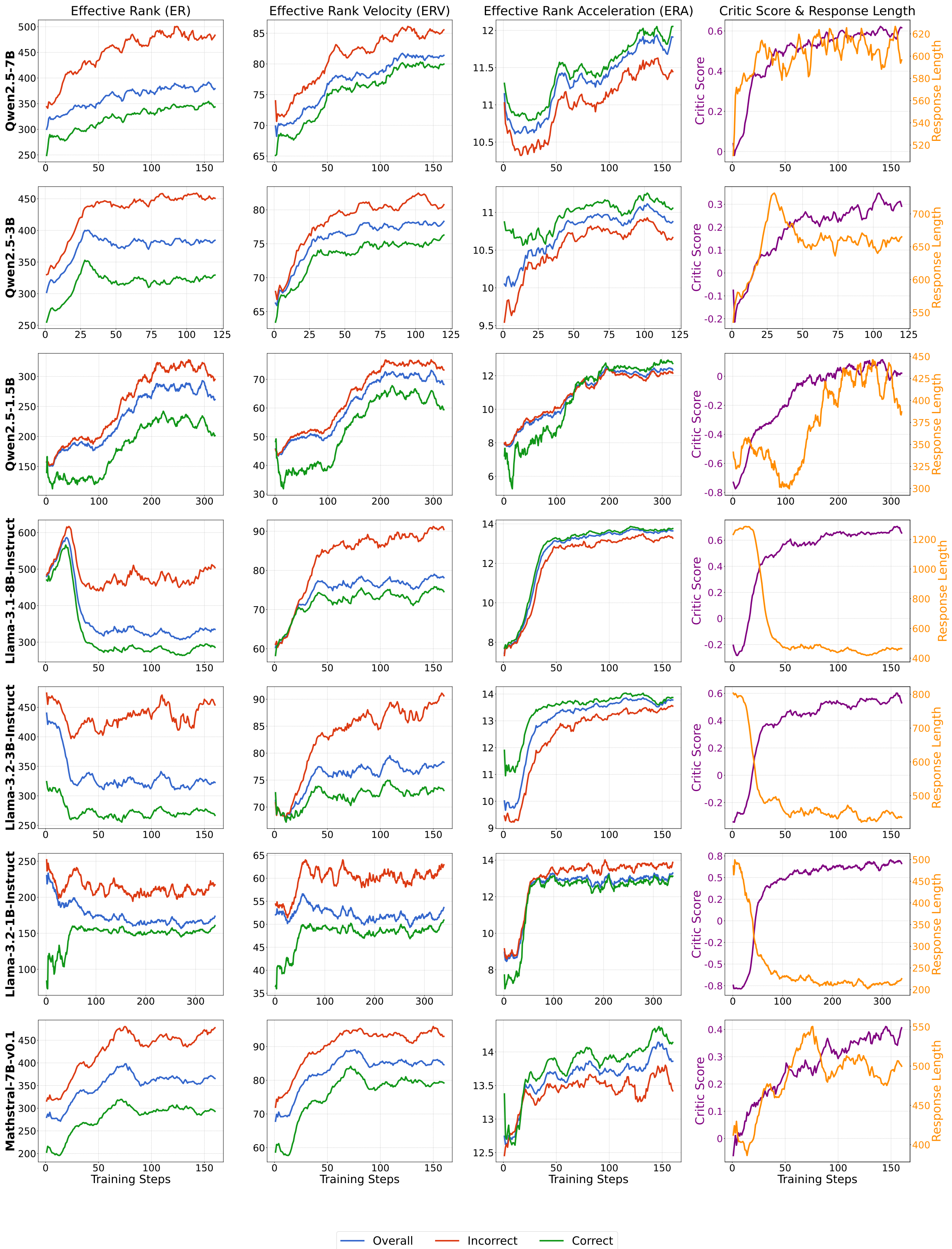


Figure 7: Response-level metrics during GRPO post-training, smoothed with a 10-step rolling window. Metrics are shown for the Overall batch, as well as for subsets of Correct and Incorrect samples. The rightmost column displays the average Critic Score (reward) and Response Length per batch.

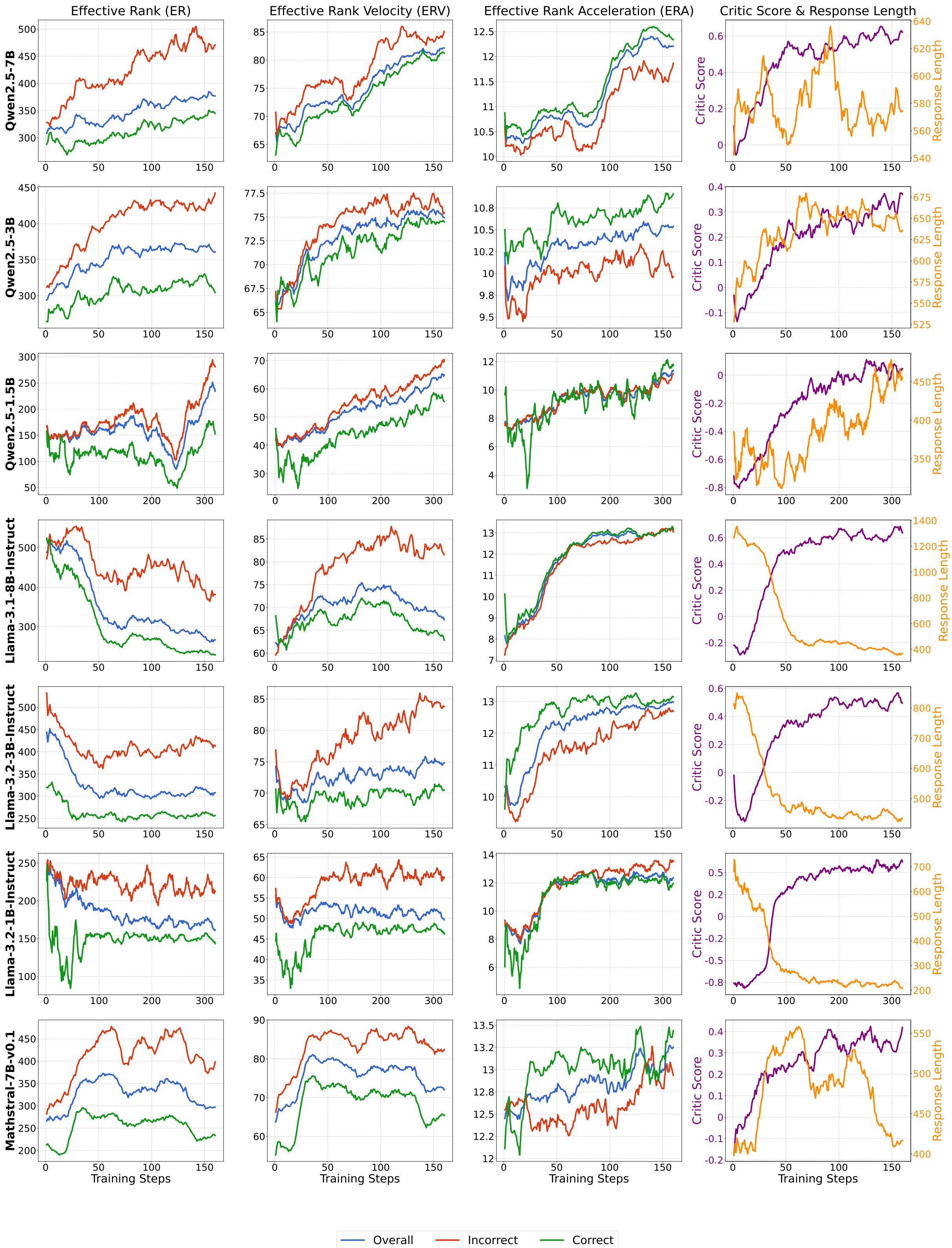


Figure 8: Response-level metrics during PPO post-training, smoothed with a 10-step rolling window. Metrics are shown for the Overall batch, as well as for subsets of Correct and Incorrect samples. The rightmost column displays the average Critic Score (reward) and Response Length per batch.

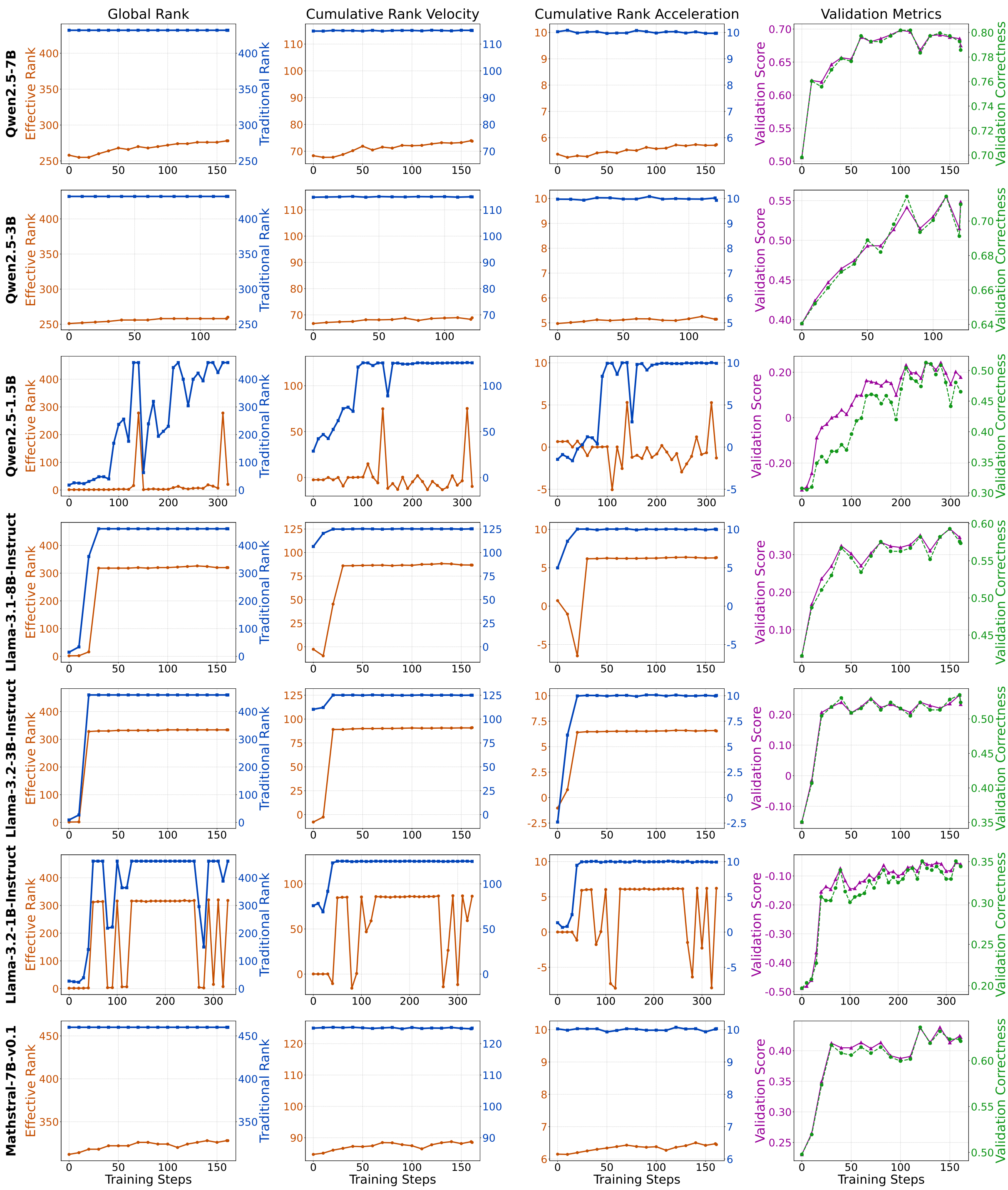


Figure 9: Visualization of dataset-level metrics during GRPO post-training. The figure compares Traditional metrics with our proposed metrics. Also shown are the Validation Score and sample Correctness, both averaged over the validation dataset.

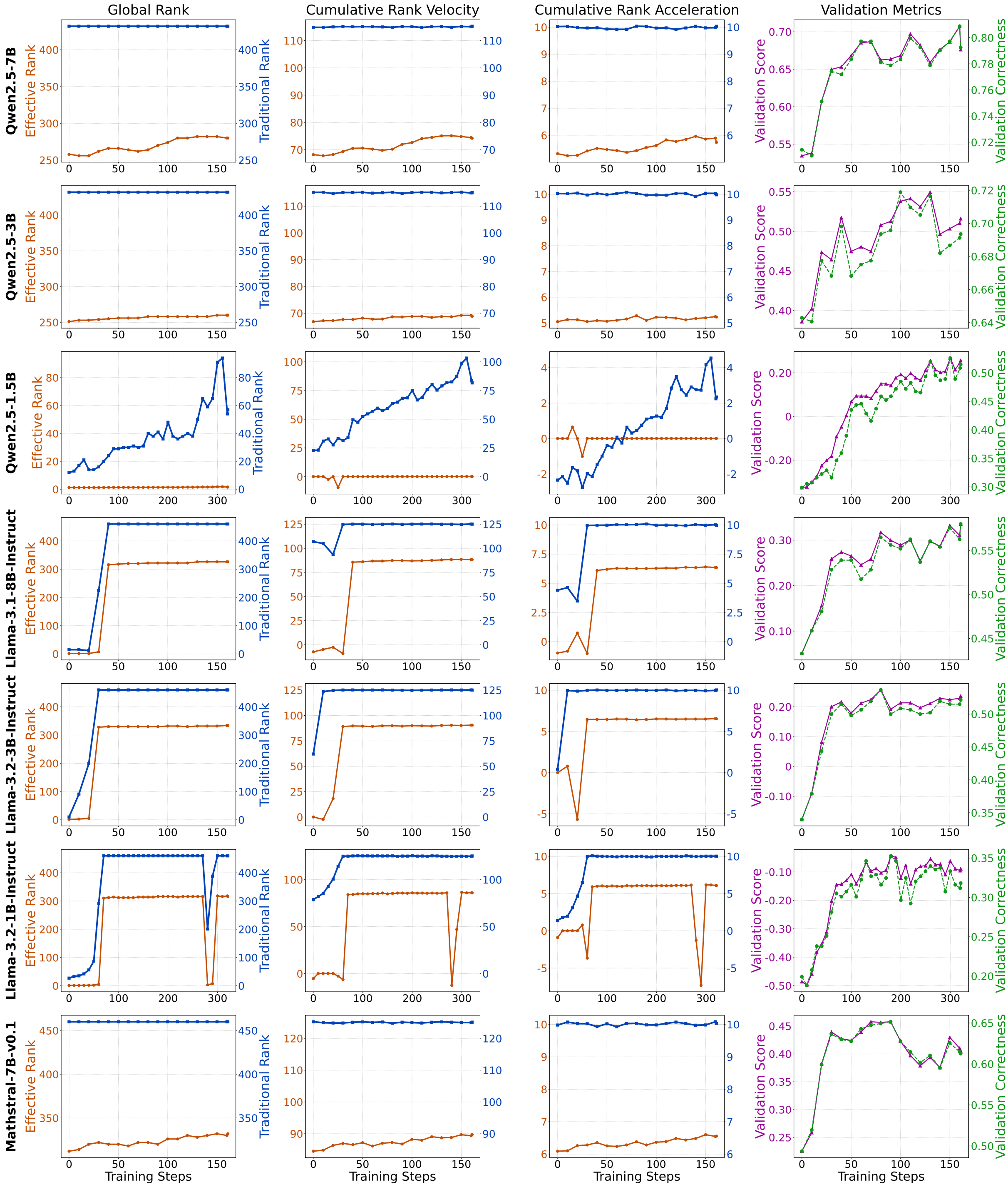


Figure 10: Visualization of dataset-level metrics during PPO post-training. The figure compares Traditional metrics with our proposed metrics. Also shown are the Validation Score and sample Correctness, both averaged over the validation dataset.

Table 4: Performance comparison of instruction-tuned models on mathematical reasoning benchmarks (Pass@1). "+ GRPO" and "+ PPO" denote reinforcement learning fine-tuning from the base model using GRPO and PPO, respectively. "w/ VERL." indicates the application of our VERL-based advantage to the corresponding RL algorithm. Δ represents the performance difference between the baseline RL method and its VERL-advanced variant. All results are reported in percentage (%).

Model	AIME24	AIME25	AMC23	AMC24	ASDiv	Carp_En	CMATH	Gaokao 2024_I	Gaokao 2024_Mix	Gaokao MathCloze	GSM8K	MAWPS	Olympiad Bench	SVAMP	TabMWP	Avg.
Llama-3.2-3B-Instruct	0.0	0.0	25.0	11.1	74.6	26.5	10.2	14.3	14.3	6.8	66.6	86.9	12.7	74.1	41.4	31.0
+ GRPO	3.3	0.0	27.5	8.9	88.8	45.0	28.3	21.4	20.9	23.7	80.7	96.0	16.7	87.7	71.7	41.4
+ GRPO w/ VERL.	13.3	6.7	25.0	11.1	89.3	45.4	46.2	14.3	22.0	22.9	81.7	96.0	17.6	87.8	72.3	43.4
Δ_{GRPO}	+10.0	+6.7	-2.5	+2.2	+0.5	+0.4	+17.9	-7.1	+1.1	-0.8	+1.0	+0.0	+0.9	+0.1	+0.6	+2.0
+ PPO	10.0	3.3	22.5	13.3	87.9	46.4	21.2	7.1	16.5	20.3	81.4	95.5	17.8	86.8	71.0	40.1
+ PPO w/ VERL.	10.0	3.3	25.0	11.1	88.7	46.0	30.7	14.3	19.8	27.1	82.9	95.7	17.3	85.8	71.3	41.9
Δ_{PPO}	+0.0	+0.0	+2.5	-2.2	+0.8	-0.4	+9.5	+7.2	+3.3	+6.8	+1.5	+0.2	-0.5	-1.0	+0.3	+1.9
Llama-3.1-8B-Instruct	0.0	3.3	17.5	8.9	48.0	34.1	18.5	0.0	15.4	16.9	47.4	43.5	10.4	48.5	34.3	23.1
+ GRPO	6.7	0.0	22.5	15.6	90.3	42.4	60.7	7.1	14.3	32.2	88.4	96.4	19.7	88.5	82.7	44.5
+ GRPO w/ VERL.	10.0	3.3	32.5	15.6	90.7	45.0	72.7	14.3	14.3	30.5	88.6	96.9	21.3	88.4	83.1	47.2
Δ_{GRPO}	+3.3	+3.3	+10.0	+0.0	+0.4	+2.6	+12.0	+7.2	+0.0	-1.7	+0.2	+0.5	+1.6	-0.1	+0.4	+2.7
+ PPO	6.7	0.0	30.0	17.8	89.8	42.0	60.0	0.0	14.3	25.4	86.4	95.7	18.2	88.6	82.3	43.8
+ PPO w/ VERL.	10.0	0.0	35.0	13.3	90.7	42.6	62.0	14.3	22.0	28.8	87.3	96.6	19.1	88.1	83.0	46.2
Δ_{PPO}	+3.3	+0.0	+5.0	-4.5	+0.9	+0.6	+2.0	+14.3	+7.7	+3.4	+0.9	+0.9	+0.9	-0.5	+0.7	+2.4
Qwen2.5-3B	6.7	0.0	20.0	24.4	90.7	54.7	76.7	0.0	22.0	41.5	80.7	95.1	23.0	84.3	71.3	46.1
+ GRPO	3.3	0.0	40.0	22.2	92.6	56.0	82.7	7.1	27.5	42.4	82.8	96.6	23.6	89.0	81.4	49.8
+ GRPO w/ VERL.	6.7	0.0	30.0	17.8	92.6	56.9	84.8	21.4	33.0	49.2	82.2	96.4	24.4	88.5	81.0	51.0
Δ_{GRPO}	+3.4	+0.0	-10.0	-4.4	+0.0	+0.9	+2.1	+14.3	+5.5	+6.8	-0.6	-0.2	+0.8	-0.5	-0.4	+1.2
+ PPO	3.3	0.0	32.5	15.6	92.8	56.5	83.2	0.0	28.6	50.0	81.7	96.6	24.4	86.0	80.8	48.8
+ PPO w/ VERL.	6.7	0.0	32.5	17.8	92.6	57.0	84.3	21.4	29.7	47.5	81.8	96.5	24.6	88.3	81.4	50.8
Δ_{PPO}	+3.4	+0.0	+0.0	+2.2	-0.2	+0.5	+1.1	+21.4	+1.1	-2.5	+0.1	-0.1	+0.2	+2.3	+0.6	+2.0
Qwen2.5-7B	6.7	0.0	45.0	15.6	91.4	55.8	86.7	42.9	33.0	49.2	85.8	95.4	25.8	88.5	82.8	53.6
+ GRPO	10.0	6.7	55.0	26.7	94.8	60.2	91.7	14.3	34.1	64.4	90.2	97.6	36.1	92.8	91.3	57.7
+ GRPO w/ VERL.	13.3	10.0	50.0	28.9	95.0	60.8	90.7	35.7	35.2	69.5	89.2	97.7	35.4	92.9	91.9	59.8
Δ_{GRPO}	+3.3	+3.3	-5.0	+2.2	+0.2	+0.6	-1.0	+21.4	+1.1	+5.1	-1.0	+0.1	-0.7	+0.1	+0.6	+2.1
+ PPO	6.7	3.3	50.0	33.3	94.9	59.6	89.8	28.6	31.9	63.6	89.1	97.3	36.1	92.8	90.8	57.9
+ PPO w/ VERL.	10.0	6.7	52.5	33.3	94.8	60.0	90.3	28.6	34.1	66.9	90.2	97.8	36.1	92.5	90.6	59.0
Δ_{PPO}	+3.3	+3.3	+2.5	+0.0	-0.1	+0.4	+0.5	+0.0	+2.2	+3.3	+1.1	+0.5	+0.0	-0.3	-0.2	+1.1
Mathstral-7B-v0.1	0.0	0.0	12.5	8.9	87.1	51.1	74.2	28.6	33.0	31.4	81.6	93.8	17.9	87.7	54.7	44.2
+ GRPO	0.0	0.0	47.5	17.8	92.9	55.9	81.3	35.7	44.0	49.2	88.1	97.6	25.6	93.0	81.5	54.0
+ GRPO w/ VERL.	6.7	0.0	45.0	20.0	93.3	55.5	81.5	50.0	40.7	46.6	89.5	97.2	29.3	90.7	83.5	55.3
Δ_{GRPO}	+6.7	+0.0	-2.5	+2.2	+0.4	-0.4	+0.2	+14.3	-3.3	-2.6	+1.4	-0.4	+3.7	-2.3	+2.0	+1.3
+ PPO	6.7	3.3	32.5	20.0	90.9	51.8	78.3	42.9	37.4	49.2	87.0	96.0	28.4	89.9	70.7	52.3
+ PPO w/ VERL.	10.0	0.0	27.5	22.2	93.0	53.8	78.2	42.9	51.6	48.3	87.4	96.7	26.1	89.6	84.1	54.1
Δ_{PPO}	+3.3	-3.3	-5.0	+2.2	+2.1	+2.0	-0.1	+0.0	+14.2	-0.9	+0.4	+0.7	-2.3	-0.3	+13.4	+1.8
Mistral-7B-v0.3	0.0	0.0	10.0	0.0	40.5	12.4	21.8	14.3	13.2	3.4	24.0	50.8	1.6	39.1	30.6	17.4
+ GRPO	0.0	0.0	2.5	4.4	58.2	11.1	42.3	0.0	15.4	5.1	52.4	79.2	3.0	47.6	37.7	23.9
+ GRPO w/ VERL.	0.0	0.0	7.5	2.2	59.1	15.0	43.0	0.0	6.6	4.2	40.3	69.5	2.8	57.5	53.0	24.0
Δ_{GRPO}	+0.0	+0.0	+5.0	-2.2	+0.9	+3.9	+0.7	+0.0	-8.8	-0.9	-12.1	-9.7	-0.2	+9.9	+15.3	+0.1
+ PPO	0.0	0.0	0.0	0.0	8.9	6.6	7.7	7.1	11.0	2.5	3.3	8.6	2.1	6.9	12.0	5.1
+ PPO w/ VERL.	0.0	0.0	2.5	0.0	44.7	10.6	35.7	7.1	16.5	5.1	28.8	70.7	2.4	57.5	35.1	21.1
Δ_{PPO}	+0.0	+0.0	+2.5	+0.0	+35.8	+4.0	+28.0	+0.0	+5.5	+2.6	+25.5	+62.1	+0.3	+50.6	+23.1	+16.0

ity. We fine-tune the base model by integrating our VERL-based Advantage method into two reinforcement learning paradigms, GRPO and PPO.

G.4 Detailed Analysis of Pass@k Performance

Tab. 5 provides a comprehensive analysis of model performance under Pass@k, a standard measure of success under larger decoding budgets. As a supplement to the main paper, it reports VERL's performance across a variety of mathematical reasoning benchmarks. These results show that VERL improves exploration under larger sampling budgets, especially on harder benchmarks.

G.5 Robustness Across Random Seeds

To assess robustness to random seed variation, we ran **five independent training seeds** for LLAMA-3.2-3B-INSTRUCT on all main benchmarks. We report mean $\pm$ standard deviation over seeds in Tab. 8 (Pass@1) and Tab. 9 (Pass@k). Compared to GRPO, adding VERL improves the **average Pass@1** from 39.24 ± 0.72 to 45.18 ± 0.38 (Tab. 8), and the **average Pass@k** from 59.87 ± 1.37 to 62.78 ± 0.63 (Tab. 9). We further conduct two-sided **Welch's t-tests** at significance level $\alpha = 0.05$ for each benchmark; statistically significant gains are marked with $\star$. Overall, most improvements remain significant, suggesting the gains are not due to random seed noise.

Most benchmarks show statistically significant gains (marked with $\star$). On the few tasks without $\star$, the mean still improves, but variance across seeds is slightly larger. On these Pass@k evaluations, GRPO+VERL again outperforms GRPO, with a statistically significant improvement in the average Pass@k (see **Avg.** column).

G.6 Detailed Analysis of OOD Performance

A key motivation of this study is to assess whether our proposed VERL algorithm improves *out-of-domain (OOD)* generalization for reasoning-centric post-training, rather than merely overfitting to the in-domain training distribution. This is aligned with our overall claim that VERL enhances robustness and OOD performance beyond standard RL

Table 5: Performance comparison of instruction-tuned models under diverse decoding settings (Pass@k). All results are reported in percentage (%).

Model	**MATH500 (Pass@16)**	**AMC23 (Pass@128)**	**AMC24 (Pass@128)**	**AIME24 (Pass@256)**	**AIME25 (Pass@256)**	**Avg.**
Llama-3.2-3B-Instruct	79.8	93.5	51.5	40.0	30.0	58.96
+ GRPO	80.2	95.4	60.6	40.0	30.0	61.24
+ GRPO w/ VERL.	80.6	95.7	59.0	50.0	36.7	64.40
Δ_{GRPO}	+0.4	+0.3	−1.6	+10.0	+6.7	+3.16
+ PPO	82.2	94.5	57.0	46.7	36.7	63.42
+ PPO w/ VERL.	82.4	94.7	57.8	46.7	40.0	64.32
Δ_{PPO}	+0.2	+0.2	+0.8	+0.0	+3.3	+0.90
Llama-3.1-8B-Instruct	79.8	94.6	57.4	46.7	36.7	63.04
+ GRPO	83.4	94.9	56.9	53.3	36.7	65.04
+ GRPO w/ VERL.	83.4	95.1	63.1	50.0	36.7	65.66
Δ_{GRPO}	+0.0	+0.2	+6.2	−3.3	+0.0	+0.62
+ PPO	79.2	92.4	59.0	46.7	36.7	62.80
+ PPO w/ VERL.	82.4	91.9	60.0	53.3	36.7	64.86
Δ_{PPO}	+3.2	−0.5	+1.0	+6.6	+0.0	+2.06
Qwen2.5-3B	86.0	96.7	69.0	56.7	40.0	69.68
+ GRPO	86.6	92.2	68.5	46.7	40.0	66.80
+ GRPO w/ VERL.	87.6	95.9	67.8	53.3	43.3	69.58
Δ_{GRPO}	+1.0	+3.7	−0.7	+6.6	+3.3	+2.78
+ PPO	87.8	96.5	67.9	43.3	43.3	67.76
+ PPO w/ VERL.	88.2	96.8	67.3	53.3	43.3	69.78
Δ_{PPO}	+0.4	+0.3	−0.6	+10.0	+0.0	+2.02
Qwen2.5-7B	90.6	98.4	73.7	60.0	60.0	76.54
+ GRPO	90.8	97.8	78.3	56.7	50.0	74.72
+ GRPO w/ VERL.	91.4	98.3	79.0	63.3	60.0	78.40
Δ_{GRPO}	+0.6	+0.5	+0.7	+6.6	+10.0	+3.68
+ PPO	91.2	98.6	74.3	53.3	56.7	74.82
+ PPO w/ VERL.	91.4	98.0	74.4	56.7	66.7	77.44
Δ_{PPO}	+0.2	−0.6	+0.1	+3.4	+10.0	+2.62
Mathstral-7B-v0.1	80.4	88.5	60.9	43.3	36.7	61.96
+ GRPO	84.8	87.3	69.2	36.7	40.0	63.60
+ GRPO w/ VERL.	87.0	97.0	76.9	50.0	50.0	72.18
Δ_{GRPO}	+2.2	+9.7	+7.7	+13.3	+10.0	+8.58
+ PPO	82.4	91.7	70.7	53.3	40.0	67.62
+ PPO w/ VERL.	84.8	93.8	69.9	53.3	46.7	69.70
Δ_{PPO}	+2.4	+2.1	−0.8	+0.0	+6.7	+2.08
Mistral-7B-v0.3	36.0	73.5	39.6	20.0	16.7	37.16
+ GRPO	33.0	63.2	36.0	10.0	10.0	30.44
+ GRPO w/ VERL.	34.4	64.5	38.0	16.7	13.3	33.38
Δ_{GRPO}	+1.4	+1.3	+2.0	+6.7	+3.3	+2.94
+ PPO	21.8	46.4	25.1	6.7	6.7	21.34
+ PPO w/ VERL.	19.2	46.5	30.1	3.3	13.3	22.48
Δ_{PPO}	−2.6	+0.1	+5.0	−3.4	+6.6	+1.14

Table 6: Comparison of GRPO + VERL using an intermediate layer (layer 14) versus the final layer, evaluated by Pass@1 on multiple math benchmarks. Using the last layer yields the strongest average improvement.

Method	aime24	aime25	amc23	amc24	asdiv	carp_en	cmath	gaokao24_I	gaokao24_mix	gaokao_math_cloze	gsm8k	mawps	olympiadbench	svamp	tabmwp	Avg.
Llama-3.2-3B-Instruct	0.0	0.0	25.0	11.1	74.6	26.5	10.2	14.3	14.3	6.8	66.6	86.9	12.7	74.1	41.4	30.97
GRPO	3.3	0.0	27.5	8.9	88.8	45.0	28.3	21.4	20.9	23.7	80.7	96.0	16.7	87.7	71.7	41.37
GRPO w/ VERL (layer = 14)	10.0	0.0	27.5	11.1	88.6	43.6	30.7	21.4	16.5	22.0	81.9	95.5	18.1	87.0	71.4	41.69
GRPO w/ VERL (layer = last)	13.3	6.7	25.0	11.1	89.3	45.4	46.2	14.3	22.0	22.9	81.7	96.0	17.6	87.8	72.3	**43.44**

Table 7: Comparison of GRPO + VERL using an intermediate layer (layer 14) versus the final layer, evaluated by Pass@k on several math benchmarks. Again, using the last layer yields the best average improvement.

Method	math500@16	amc23@128	amc24@128	aime24@256	aime25@256	Avg.
Llama-3.2-3B-Instruct	79.8	93.5	51.5	40.0	30.0	58.96
GRPO	80.2	95.4	60.6	40.0	30.0	61.24
GRPO w/ VERL (layer = 14)	81.0	92.9	57.0	40.0	36.7	61.52
GRPO w/ VERL (layer = last)	80.6	95.7	59.0	50.0	36.7	**64.40**

Table 8: Seed robustness on main benchmarks (Pass@1). Results are mean $\pm$ std over 5 seeds. $\star$ indicates Welch's t-test significance at $\alpha = 0.05$ comparing GRPO+VERL against GRPO.

Method	aime24	aime25	amc23	amc24	asdiv	carp_en	cmath	gaokao24_I	gaokao24_mix	gaokao_cloze	gsm8k	mawps	olympiad	svamp	tabmwp	**Avg.**
GRPO	10.00 ± 0.00	0.00 ± 0.00	17.50 ± 0.00	11.54 ± 0.98	88.62 ± 0.54	44.82 ± 0.04	15.34 ± 0.36	5.72 ± 7.83	19.58 ± 2.95	22.20 ± 1.82	80.88 ± 0.60	95.60 ± 0.00	17.80 ± 0.85	87.08 ± 0.22	71.92 ± 0.41	$\mathbf{39.24 \pm 0.72}$
GRPO+VERL	$12.64 \pm 1.48\star$	$3.98 \pm 1.52\star$	$25.50 \pm 1.12\star$	$16.00 \pm 2.46\star$	$89.50 \pm 0.14\star$	$45.20 \pm 0.14\star$	$56.62 \pm 0.93\star$	$25.72 \pm 3.94\star$	18.92 ± 1.97	24.04 ± 0.76	$83.00 \pm 0.59\star$	$96.32 \pm 0.16\star$	18.30 ± 0.71	87.30 ± 0.28	$74.64 \pm 0.75\star$	$\mathbf{45.18 \pm 0.38}\star$

Table 9: Seed robustness on harder Pass@k settings. Results are mean $\pm$ std over 5 seeds. $\star$ indicates Welch's t-test significance at $\alpha = 0.05$.

Method	math500@16	amc23@128	amc24@128	aime24@256	aime25@256	**Avg.**
GRPO	80.40 ± 0.00	90.78 ± 0.18	56.18 ± 1.50	39.34 ± 1.48	32.66 ± 5.95	$\mathbf{59.87 \pm 1.37}$
GRPO+VERL	80.52 ± 0.18	$94.22 \pm 1.01\star$	58.46 ± 1.69	$43.98 \pm 3.66\star$	36.70 ± 0.00	$\mathbf{62.78 \pm 0.63}\star$

Table 10: Out-of-domain (OOD) reasoning performance (Pass@1, %) across diverse domains. Results are reported for Mathstral-7B-v0.1 and Qwen2.5-7B, comparing vanilla GRPO/PPO and their VERL-enhanced variants.

Model / Method	**AstroQA EN**	**AstroQA ZH**	**GPQA D**	**GPQA Ext**	**GPQA Main**	**SciBench**	**LEXam**	**LexEval**	**LogiQA EN**	**LogiQA ZH**	**ReClor**	**MedMCQA**	**MedQA**	**PubMedQA**	**MMLU**	**MMLU-Pro**	**Avg.**
Mathstral-7B-v0.1	50.7	43.8	29.3	26.0	25.4	15.8	19.6	38.7	45.8	44.6	43.6	42.9	39.7	22.4	53.9	26.4	35.25
+ GRPO	57.1	49.3	28.8	29.3	29.9	22.4	22.4	42.8	47.1	45.7	46.3	44.9	43.8	22.6	56.7	29.8	39.31
+ GRPO w/ VERL.	58.0	51.6	31.8	31.1	34.6	27.9	22.6	47.6	50.4	44.8	49.7	47.1	46.1	24.4	58.8	30.7	41.68
Δ_{GRPO}	+0.9	+2.3	+3.0	+1.8	+4.7	+5.5	+0.2	+4.8	+3.3	−0.9	+3.4	+2.2	+2.3	+1.8	+2.1	+0.9	+2.37
+ PPO	52.7	47.5	28.8	29.9	27.9	22.1	17.3	44.5	47.5	45.0	46.5	44.2	45.2	23.1	56.5	29.1	37.93
+ PPO w/ VERL.	54.5	49.1	31.3	26.6	26.1	23.0	21.4	42.2	47.5	51.6	46.2	44.8	45.1	22.3	56.9	29.3	39.34
Δ_{PPO}	+1.8	+1.6	+2.5	−3.3	−1.8	+0.9	+4.1	−2.3	+0.0	+6.6	−0.3	+0.6	−0.1	−0.8	+0.4	+0.2	+1.42
Qwen2.5-7B	61.2	58.0	27.3	26.6	24.6	15.8	21.1	50.4	50.8	49.7	56.0	50.6	62.0	43.6	61.3	36.1	43.44
+ GRPO	59.6	61.3	27.3	27.8	28.8	21.5	22.0	53.6	52.1	52.1	58.8	52.1	69.2	46.6	60.6	33.8	46.15
+ GRPO w/ VERL.	59.4	62.4	30.8	28.9	26.8	22.8	23.0	54.6	52.8	52.4	61.6	52.0	70.3	47.8	60.9	33.4	47.09
Δ_{GRPO}	−0.2	+1.1	+3.5	+1.1	−2.0	+1.3	+1.0	+1.0	+0.7	+0.3	+2.8	−0.1	+1.1	+1.2	+0.3	−0.4	+0.94
+ PPO	60.0	62.1	27.3	25.3	28.3	22.1	22.8	55.2	52.1	52.1	58.6	53.6	70.4	45.8	60.9	34.1	46.20
+ PPO w/ VERL.	60.9	61.3	34.8	30.0	27.0	23.0	23.4	57.1	53.0	52.3	61.8	52.9	71.0	45.4	61.0	34.4	47.45
Δ_{PPO}	+0.9	−0.8	+7.5	+4.7	−1.3	+0.9	+0.6	+1.9	+0.9	+0.2	+3.2	−0.7	+0.6	−0.4	+0.1	+0.3	+1.25

fine-tuning baselines.

To stress-test OOD generalization, we evaluate on a diverse suite of reasoning benchmarks spanning multiple domains: (i) **Astronomy**: AstroQA (English/Chinese) (Li et al., 2025a); (ii) **Science**: GPQA (Diamond/Extended/Main) (Rein et al., 2024) and SciBench (Wang et al., 2024b); (iii) **Law**: LEXam (Fan et al., 2025) and LexEval (Li et al., 2024); (iv) **Logical reasoning**: LogiQA (English/Chinese) (Liu et al., 2020) and ReClor (Yu et al., 2020); (v) **Medical QA**: MedMCQA (Pal et al., 2022), MedQA (Jin et al., 2021), and PubMedQA (Jin et al., 2019); (vi) **Broad knowledge**: MMLU (Hendrycks et al., 2021a) and MMLU-Pro (Wang et al., 2024c). All results are reported as Pass@1 (%).

Tab. 10 summarizes the OOD Pass@1 results for both Mathstral-7B and Qwen2.5-7B. Overall, VERL yields consistent average improvements when plugged into either GRPO or PPO: for Mathstral-7B, VERL improves the OOD average from 39.31 to 41.68 (+2.37) on GRPO and from 37.93 to 39.34 (+1.42) on PPO; for Qwen2.5-7B, it improves the OOD average from 46.15 to 47.09 (+0.94) on GRPO and from 46.20 to 47.45 (+1.25) on PPO. These gains are broadly distributed across heterogeneous domains, supporting that VERL enhances OOD reasoning robustness rather than trading off generalization for in-domain gains.

Table 11: Comparison with action-space regularization baselines on selected reasoning datasets (Pass@1, %). Best value in each column is in **bold**; second best is underlined.

Model / Method	AIME24	AIME25	AMC23	AMC24	ASDiv	CARP_EN	CMath	Gaokao24_I	Gaokao24_mix	Gaokao_math _cloze	GSM8K	MAWPS	OlympiadBench	SVAMP	TabMWP	Avg.
Llama-3.2-3B-Instruct	0.0	0.0	25.0	11.1	74.6	26.5	10.2	14.3	14.3	6.8	66.6	86.9	12.7	74.1	41.4	30.97
+ GRPO	3.3	0.0	27.5	8.9	88.8	45.0	28.3	**21.4**	20.9	23.7	80.7	96.0	16.7	87.7	71.7	41.37
+ GRPO w/ ACTION-LOGP regu	3.3	0.0	27.5	**15.6**	88.8	44.6	38.2	7.1	15.4	22.0	80.9	**96.1**	16.3	87.4	71.9	41.01
+ GRPO w/ ACTION-ENTROPY regu	10.0	0.0	**35.0**	8.9	89.2	45.3	37.5	7.1	16.5	**28.0**	81.9	**96.1**	**17.9**	87.0	71.6	42.13
+ GRPO w/ ACTION-LOGP+ENTROPY	3.3	3.3	25.0	13.3	88.5	44.7	37.7	0.0	18.7	23.7	**82.0**	**96.1**	15.6	87.0	70.5	40.63
+ GRPO w/ VERL (ours)	**13.3**	**6.7**	25.0	11.1	**89.3**	**45.4**	**46.2**	14.3	**22.0**	22.9	81.7	96.0	17.6	**87.8**	**72.3**	**43.40**

Table 12: Comparison with action-space regularization baselines under larger decoding budgets (Pass@k, %). Best value in each column is in **bold**; second best is underlined.

Model / Method	Math500 (Pass@16)	AMC23 (Pass@128)	AMC24 (Pass@128)	AIME24 (Pass@256)	AIME25 (Pass@256)	Avg.
Llama-3.2-3B-Instruct	79.8	93.5	51.5	40.0	30.0	58.96
+ GRPO	80.2	95.4	**60.6**	40.0	30.0	61.24
+ GRPO w/ ACTION-LOGP regu	79.6	93.8	56.9	40.0	30.0	60.06
+ GRPO w/ ACTION-ENTROPY regu	78.4	93.1	56.9	46.7	**36.7**	62.36
+ GRPO w/ ACTION-LOGP+ENTROPY	80.0	93.9	55.8	40.0	**36.7**	61.28
+ GRPO w/ VERL (ours)	**80.6**	**95.7**	59.0	**50.0**	**36.7**	**64.40**

G.7 Comparison with Regularization Method in Action Space

This experiment aims to compare VERL with prior *action-space* regularization strategies used in RL post-training. Here, "action space" refers to the token-level policy over next-token actions during generation. A common practice is to regularize training using token-level proxy signals computed directly from this policy (e.g., confidence or entropy). Our goal is to test whether such action-space proxy regularizers can match the performance gains of VERL, and to provide evidence that representation-level metrics yield a more effective regularization signal for reasoning.

Setup. We keep the post-training pipeline unchanged (same base model, RL algorithm, training data, optimizer, and decoding configuration), and only replace the auxiliary regularization signal. We consider three action-space baselines: (i) ACTION-LOGP, which regularizes using only the **average log-probability** of the generated response (a confidence-style signal); (ii) ACTION-ENTROPY, which regularizes using only the **response entropy** (a stochasticity-style signal); (iii) ACTION-LOGP+ENTROPY, which uses the **sum** of the above two signals. In contrast, VERL (VERL) employs hidden-state trajectory metrics defined in a semantically richer representation space. We report Pass@1 results in Tab. 11 and larger-budget decoding results (Pass@k) in Tab. 12.

Findings. Two observations stand out. First, action-space proxy regularization is often *unstable* under combination: while a single proxy can sometimes help (e.g., ACTION-ENTROPY achieves the strongest or second-strongest performance on several columns), the naive combination ACTION-LOGP+ENTROPY does not reliably improve further and can even underperform the better single-signal choice (see the Avg. column in Tab. 11 and the Pass@k columns in Tab. 12). This indicates that token-level confidence (LogP) and stochasticity (entropy) may interfere when jointly used as a regularizer.

Second, across both evaluation regimes, VERL consistently provides the best overall results. On Pass@1, VERL achieves the highest Avg. and attains the best (or second-best) scores on most datasets in Tab. 11; on Pass@k, VERL similarly yields the strongest Avg. and the top performance on key high-budget reasoning benchmarks (e.g., AIME24 at Pass@256 and AMC23 at Pass@128 in Tab. 12). Overall, these results suggest that action-space token-level proxies are limited and do not reliably compose, whereas VERL offers a more robust and effective regularization signal for RL post-training.

G.8 Ablation on the Choice of Hidden Layer

We focus on the final hidden layer because our exploration/exploitation metrics are defined in the *semantic* space along the reasoning trajectory, and prior interpretability work (Jing et al., 2025; Sajjad et al., 2022; Valeriani et al., 2023; Matthews et al., 2024; Servedio et al., 2025; Zhang et al., 2025a) suggests that the last layers are most aligned with semantic meaning and model predictions. Empirically, using the final layer gives consistently better performance than using an intermediate layer: for example, GRPO + VERL with the last layer improves the average Pass@1 from 41.69% (layer 14) to 43.44%, and the average Pass@k from 61.52% (layer 14) to 64.40%. This subsection provides the detailed analysis supporting our design choice to base VERL on the final layer.

Intermediate layers in large language models can encode rich features. However, our notion of exploration and exploitation is explicitly defined in the semantic space of a reasoning trajectory. Existing interpretability studies (Jing et al., 2025; Sajjad et al., 2022; Valeriani et al., 2023; Matthews et al., 2024; Servedio et al., 2025; Zhang et al., 2025a) indicate that hidden states in the last layers are most tightly aligned with token-level semantics and the model's predictive distribution, while mid-layer representations tend to mix morphology, syntax, and other lower-level or task-specific signals. For this reason, we consider the final layer more suitable for semantic diversity metrics.

Intermediate-layer vs. final-layer VERL (Pass@1). We use Llama-3.2-3B-Instruct as the base model, which has 28 transformer layers. We implemented VERL on an intermediate layer (layer 14) and on the last layer, keeping everything else fixed. The Pass@1 results are reported in Tab. 6. Both VERL variants improve over GRPO, but the last-layer version clearly gives the strongest overall gains in average Pass@1.

Intermediate-layer vs. final-layer VERL (Pass@k). A similar pattern holds for Pass@k, summarized in Tab. 7. Using the last layer yields the best average improvement, especially on the more challenging AIME-style benchmarks, further supporting the choice of the final hidden layer as the basis for our semantic exploration and exploitation metrics.

H Case Study

All case studies use Qwen2.5-7B as the base model. Outputs in the **gray boxes** are produced by vanilla GRPO trained for 120 steps, whereas outputs in the **purple boxes** are produced by VERL (GRPO + auxiliary shaping) trained for the same 120 steps.

H.1 Case Study For Pass@1 Setting

Case Study I. For the case in Fig. 11, vanilla GRPO incorrectly concludes that $-1 < -13$, while GRPO with our auxiliary shaping term produces the correct comparison. This illustrates a weakness in vanilla GRPO's exploitation: it does not reliably apply basic numerical reasoning, namely that among negative numbers, a larger absolute value implies a smaller number. Adding the auxiliary shaping term improves exploitation, making such comparisons more consistent.

Case Study II. For the case in Fig. 12, the crucial distinction lies in how the models interpret the condition *"No house faces two different streets"*. The vanilla GRPO–trained model incorrectly infers that each street contains only 10 houses, overlooking the structural implication of the condition. In contrast, the model trained with GRPO augmented by the auxiliary shaping term exploits this commonsense constraint more effectively: it decomposes the problem step by step, correctly deduces that each street has 20 houses, and thereby arrives at the correct total.

Case Study III. As shown in Fig. 13, the task concerns solving a differential equation under the constraint of finding a purely exponential solution rather than a general one. The model trained with vanilla GRPO overlooks this constraint and defaults to the generic procedure for obtaining the full solution. In contrast, the model trained with GRPO augmented by the auxiliary shaping term respects the constraint and produces the correct purely exponential answer. This case illustrates how the auxiliary shaping term enhances exploitation by enforcing adherence to problem-specific constraints.

H.2 Case Study For Pass@16 Setting

Case Study I. As shown in Fig. 14, the question is a simple elementary-level math problem requiring the integer solution $x \geqslant 1, x \neq 2$. While the vanilla GRPO model consistently outputs 3, which is incorrect, the model trained with the auxiliary shaping term produces 1 in 5 out of 16 rollouts, the correct answer. This case highlights that the auxiliary shaping term enhances the model's exploration ability, enabling it to consider a wider range of possibly valid solutions.

Case Study II. As shown in Fig. 15, this high-school level geometry problem illustrates the contrast between vanilla GRPO and GRPO with the auxiliary shaping term. The vanilla GRPO model fails to produce the correct answer in any rollout. In contrast, the model trained with the auxiliary shaping term succeeds in 5 out of 16 rollouts. Moreover, its reasoning traces display higher ex-

Question

What is the smallest possible integer value of x if $12 \div (x+1)$ is an integer?

Ground Truth

-13

Vanilla GRPO

To find the smallest possible integer value of x such that $\frac{12}{x+1}$ is an integer, we need to consider the factors of 12. The expression $\frac{12}{x+1}$ will be an integer if $x+1$ is a divisor of 12. Let's list the divisors of 12:
1. The divisors of 12 are: $\pm 1, \pm 2, \pm 3, \pm 4, \pm 6, \pm 12$.
2. We need to find the smallest x, so we start with the smallest positive divisor, which is 1:

$$x + 1 = 1 \implies x = 1 - 1 \implies x = 0$$

3. To confirm, we substitute $x = 0$ back into the original expression:

$$\frac{12}{0+1} = \frac{12}{1} = 12$$

Since 12 is an integer, $x = 0$ is indeed a valid solution.
4. We should also consider the negative divisors to ensure we have the smallest possible integer value. The next smallest divisor is -1:

$$x + 1 = -1 \implies x = -1 - 1 \implies x = -2$$

5. Substituting $x = -2$ back into the original expression:

$$\frac{12}{-2+1} = \frac{12}{-1} = -12$$

Since -12 is an integer, $x = -2$ is also a valid solution, but it is not smaller than $x = 0$.
Therefore, the smallest possible integer value of x is

$$\boxed{0}$$

GRPO with Auxiliary Shaping Term

To find the smallest possible integer value of x such that $\frac{12}{x+1}$ is an integer, we need to determine the divisors of 12. The divisors of 12 are $\pm 1, \pm 2, \pm 3, \pm 4, \pm 6, \pm 12$. For $\frac{12}{x+1}$ to be an integer, $x+1$ must be one of these divisors. We want the smallest possible x, so we should consider the smallest positive divisor, which is 1.
If $x + 1 = 1$, then $x = 0$.
We should also consider the negative divisors to ensure we find the smallest possible integer value of x. The smallest negative divisor is -12. If $x + 1 = -12$, then $x = -13$.
Comparing $x = 0$ and $x = -13$, the smallest possible integer value of x is -13.
Therefore, the final answer is

$$\boxed{-13}$$

Figure 11: Case study: Vanilla GRPO fails to exploit numerical commonsense when comparing negative values (-1 vs. -13), while our method gives the correct answer.

ploratory diversity: it applies the law of sines 9 times, the law of cosines 4 times, and other general geometric properties 3 times. By comparison, the vanilla GRPO model relies more narrowly on the law of sines (11 times), with only limited use of other strategies. This case highlights that the auxiliary shaping term enhances the model's exploration capacity, enabling it to leverage a broader range of problem-solving strategies rather than over-committing to a single path.

Question

In my neighborhood, there are six streets. There are 10 houses on each side of each street. No house faces two different streets. How many houses are in my neighborhood?

Ground Truth

120

Vanilla GRPO

To determine the total number of houses in the neighborhood, we can follow these steps:
1. **Identify the number of streets and houses per street:** - There are 6 streets. - Each street has 10 houses on each side.
2. **Calculate the number of houses on one street:** Since there are 10 houses on each side of a street, and each house faces only one street, the total number of houses on one street is 10 (since they are all on the same street).
3. **Calculate the total number of houses in the neighborhood:** There are 6 streets, and each street has 10 houses. Therefore, the total number of houses is $6 \times 10 = 60$.
Putting the final answer within $\boxed{}$, we get:

$$\boxed{60}$$

GRPO with Auxiliary Shaping Term

To find the total number of houses in the neighborhood, let's break down the problem step by step.
1. There are 6 streets in the neighborhood.
2. Each street has houses on both sides.
3. There are 10 houses on each side of every street.
4. No house faces two different streets, meaning each house is counted only once.
First, calculate the number of houses on one side of a single street:

$$10 \text{ houses/side}$$

Since there are houses on both sides of each street, multiply by 2:

$$10 \text{ houses/side} \times 2 \text{ sides/street} = 20 \text{ houses/street}$$

Now, since there are 6 streets, multiply the number of houses per street by the number of streets:

$$20 \text{ houses/street} \times 6 \text{ streets} = 120 \text{ houses}$$

So, the total number of houses in the neighborhood is

$$\boxed{120}$$

Figure 12: Case study: Vanilla GRPO overlooks the constraint and infers 10 houses per street, while our method exploits it correctly and infers 20.

Question

Find a purely exponential solution of $\frac{d^4x}{dt^4} - x = e^{-2t}$.

Ground Truth

$\frac{e^{-2t}}{15}$

Vanilla GRPO

To find a purely exponential solution of the differential equation ($\frac{d^4x}{dt^4} - x = e^{-2t}$), we will follow these steps:
1. **Find the complementary solution by solving the homogeneous equation:**

$$\frac{d^4x}{dt^4} - x = 0$$

The characteristic equation is:

$$r^4 - 1 = 0$$

Solving for r, we get:

$$r^4 = 1 \implies r = \pm 1, \pm i$$

Therefore, the complementary solution is:

$$x_c(t) = C_1 e^t + C_2 e^{-t} + C_3 \cos(t) + C_4 \sin(t)$$

2. **Find a particular solution using the method of undetermined coefficients:**
Since the right-hand side of the non-homogeneous equation is e^{-2t}, we assume a particular solution of the form:

$$x_p(t) = Ae^{-2t}$$

Taking the fourth derivative of $x_p(t)$:

$$\frac{d^4}{dt^4}(Ae^{-2t}) = A(-2)^4 e^{-2t} = 16Ae^{-2t}$$

Substituting $x_p(t)$ and its fourth derivative into the original differential equation:

$$16Ae^{-2t} - Ae^{-2t} = e^{-2t}$$

Simplifying, we get:

$$(16A - A)e^{-2t} = e^{-2t} \implies 15Ae^{-2t} = e^{-2t}$$

Therefore:

$$15A = 1 \implies A = \frac{1}{15}$$

So, the particular solution is:

$$x_p(t) = \frac{1}{15} e^{-2t}$$

3. **Combine the complementary and particular solutions:**
The general solution to the differential equation is:

$$x(t) = x_c(t) + x_p(t) = C_1 e^t + C_2 e^{-t} + C_3 \cos(t) + C_4 \sin(t) + \frac{1}{15} e^{-2t}$$

Thus, the final answer is:

$$\boxed{C_1 e^t + C_2 e^{-t} + C_3 \cos(t) + C_4 \sin(t) + \frac{1}{15} e^{-2t}}$$

Figure 13: Case study: Vanilla GRPO ignores the constraint of a purely exponential solution, while our method enforces it and yields the correct result.

GRPO with Auxiliary Shaping Term

To find a purely exponential solution to the differential equation

$$\frac{d^4x}{dt^4} - x = e^{-2t}$$

, we will follow these steps:

1. **Find the complementary solution by solving the homogeneous equation:**

$$\frac{d^4x}{dt^4} - x = 0$$

The characteristic equation is:

$$r^4 - 1 = 0$$

Solving for r, we get:

$$r^4 = 1 \implies r = \pm 1, \pm i$$

Therefore, the complementary solution is:

$$x_c(t) = C_1e^t + C_2e^{-t} + C_3\cos(t) + C_4\sin(t)$$

2. **Find a particular solution using the method of undetermined coefficients:** Since the right-hand side of the non-homogeneous equation is e^{-2t}, we assume a particular solution of the form:

$$x_p(t) = Ae^{-2t}$$

Taking the fourth derivative of $x_p(t)$:

$$\frac{d^4x_p}{dt^4} = 16Ae^{-2t}$$

Substituting $x_p(t)$ and its fourth derivative into the original differential equation:

$$16Ae^{-2t} - Ae^{-2t} = e^{-2t}$$

Simplifying, we get:

$$15Ae^{-2t} = e^{-2t}$$

Solving for A:

$$15A = 1 \implies A = \frac{1}{15}$$

Therefore, the particular solution is:

$$x_p(t) = \frac{1}{15}e^{-2t}$$

3. **Combine the complementary and particular solutions:** The general solution to the differential equation is:

$$x(t) = x_c(t) + x_p(t) = C_1e^t + C_2e^{-t} + C_3\cos(t) + C_4\sin(t) + \frac{1}{15}e^{-2t}$$

Thus, the purely exponential solution is:

$$\boxed{\frac{1}{15}e^{-2t}}$$

Figure 13: Case study: Vanilla GRPO ignores the constraint of a purely exponential solution, while our method enforces it and yields the correct result (continued).

Question

Given $f(x) = \frac{\sqrt{x-1}}{x-2}$, what is the smallest possible integer value for x such that $f(x)$ has a real number value?

Ground Truth

1

Figure 14: Case study demonstrating improved exploration: the auxiliary shaping term enables the model to explore and reach the correct solution.

Vanilla GRPO#1-#8

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The numerator is $\sqrt{x-1}$. For the square root to be defined and result in a real number, the expression inside the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator is $x - 2$. For the function to be defined, the denominator must not be zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

To determine the smallest possible integer value for x such that $f(x) = \frac{\sqrt{x-1}}{x-2}$ has a real number value, we need to ensure that both the numerator and the denominator satisfy their respective conditions for real numbers.
1. **Numerator Condition**: The expression inside the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator must not be zero to avoid division by zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these conditions, we need $x \geqslant 1$ and $x \neq 2$. The smallest integer that satisfies these conditions is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

To determine the smallest possible integer value for x such that $f(x) = \frac{\sqrt{x-1}}{x-2}$ has a real number value, we need to ensure that both the numerator and the denominator satisfy certain conditions.
1. **Numerator Condition**: The expression inside the square root must be non-negative because the square root of a negative number is not a real number.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator must not be zero because division by zero is undefined.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.

(Details omitted as they are not crucial for this case study.)
1. **Square Root Condition**: The expression inside the square root, $x - 1$, must be greater than or equal to zero for the square root to be a real number.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero to avoid division by zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. For the square root to be defined, the expression inside it must be greater than or equal to zero:

$$x - 1 \geqslant 0$$

Solving this inequality:

$$x \geqslant 1$$

2. For the denominator to be non-zero:

$$x - 2 \neq 0$$

Solving this equation:

$$x \neq 2$$

(Details omitted as they are not crucial for this case study.)
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is:

$$\boxed{3}$$

(Details omitted as they are not crucial for this case study.)
1. For the square root to be defined, the expression inside it must be greater than or equal to zero:

$$x - 1 \geqslant 0$$

Solving this inequality:

$$x \geqslant 1$$

2. For the function to be defined, the denominator must not be zero:

$$x - 2 \neq 0$$

Solving this equation:

$$x \neq 2$$

(Details omitted as they are not crucial for this case study.)
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. For the square root to be defined, we need:

$$x - 1 \geqslant 0$$

Solving this inequality:

$$x \geqslant 1$$

2. For the denominator to be non-zero, we need:

$$x - 2 \neq 0$$

Solving this equation:

$$x \neq 2$$

Combining these two conditions, x must be greater than or equal to 1 but not equal to 2. The smallest integer that satisfies this condition is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. **Square Root Condition**: The expression inside the square root, $x - 1$, must be greater than or equal to zero for the square root to be a real number.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero to avoid division by zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

Figure 14: Case study demonstrating improved exploration: the auxiliary shaping term enables the model to explore and reach the correct solution (continued).

Vanilla GRPO#9-#16

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression inside the square root, $x - 1$, must be non-negative because the square root of a negative number is not a real number. Therefore, we need:

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero because division by zero is undefined. Therefore, we need:

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is: $\boxed{3}$

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression inside the square root, $x - 1$, must be non-negative because the square root of a negative number is not a real number. Therefore, we need:

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero because division by zero is undefined. Therefore, we need:

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. For the square root to be defined, we need:

$$x - 1 \geqslant 0$$

$$x \geqslant 1$$

2. For the denominator to be non-zero, we need:

$$x - 2 \neq 0$$

$$x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1$$

$$x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression under the square root, $x - 1$, must be non-negative because the square root of a negative number is not a real number. Therefore, we need:

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero because division by zero is undefined. Therefore, we need:

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is: $\boxed{3}$

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The numerator is $\sqrt{x-1}$. For the square root to be defined and result in a real number, the expression inside the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator is $x - 2$. For the function to be defined, the denominator must not be zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we need x to be greater than or equal to 1, but not equal to 2. Therefore, the smallest possible integer value for x that satisfies both conditions is 3.
$\boxed{3}$

(Details omitted as they are not crucial for this case study.)
1. **Square Root Condition**: The expression inside the square root, $x - 1$, must be greater than or equal to zero for the square root to be a real number.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero to avoid division by zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

To determine the smallest possible integer value for x such that $f(x) = \frac{\sqrt{x-1}}{x-2}$ has a real number value, we need to ensure that both the numerator and the denominator satisfy their respective conditions for real numbers.
1. **Numerator Condition**: The expression inside the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator must not be zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these conditions, we need x to be greater than or equal to 1 but not equal to 2. Therefore, the smallest possible integer value for x that satisfies both conditions is 3.
$\boxed{3}$

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The numerator is $\sqrt{x-1}$. For the square root to be a real number, the expression inside the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator is $x - 2$. For the function to be defined, the denominator must not be zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

Figure 14: Case study demonstrating improved exploration: the auxiliary shaping term enables the model to explore and reach the correct solution (continued).

GRPO with Auxiliary Shaping Term#1-#8

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The numerator is $\sqrt{x-1}$. For the square root to be defined and real, the expression inside the square root must be non-negative.

$$x-1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator is $x-2$. For the function to be defined, the denominator must not be zero.

$$x-2 \neq 0 \implies x \neq 2$$

Combining these conditions, we need x to be greater than or equal to 1, but not equal to 2. The smallest integer that satisfies these conditions is $x=3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. For the square root to be defined in the real numbers, the expression inside it must be greater than or equal to zero:

$$x-1 \geqslant 0$$

Solving this inequality:

$$x \geqslant 1$$

2. For the function to be defined, the denominator must not be zero:

$$x-2 \neq 0$$

Solving this equation:

$$x \neq 2$$

(Details omitted as they are not crucial for this case study.)
Thus, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. The expression inside the square root must be non-negative. 2. The denominator must not be zero.
(Details omitted as they are not crucial for this case study.)
Step 3: Combine the conditions From Step 1, we know $x \geqslant 1$. From Step 2, we know $x \neq 2$. Therefore, the smallest integer value for x that satisfies both conditions is the smallest integer greater than 1, which is 3.
Final Answer The smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
Combining the Conditions From Condition 1, we have $x \geqslant 1$. From Condition 2, we have $x \neq 2$.
Combining these, the smallest integer value for x that satisfies both conditions is $x=3$.
Verification Let's verify that $x=3$ works: - For $x=3$, $\sqrt{x-1}=\sqrt{3-1}=\sqrt{2}$, which is a real number. - The denominator $x-2=3-2=1$, which is not zero.
Since both conditions are satisfied, the smallest possible integer value for x is indeed 3.
Thus, the final answer is:

$$\boxed{3}$$

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression under the square root, $x-1$, must be non-negative because the square root of a negative number is not a real number. Therefore, we need:

$$x-1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x-2$, must not be zero because division by zero is undefined. Therefore, we need:

$$x-2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ but not $x=2$ is $x=1$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{1}$.

(Details omitted as they are not crucial for this case study.)
Combining the Conditions From Condition 1, we know $x \geqslant 1$. From Condition 2, we know $x \neq 2$.
The smallest integer that satisfies $x \geqslant 1$ but not $x=2$ is $x=1$.
(Details omitted as they are not crucial for this case study.)
Since $f(1)$ is a real number, $x=1$ is indeed a valid solution.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{1}$.

(Details omitted as they are not crucial for this case study.)
1. For $\sqrt{x-1}$ to be defined, we need:

$$x-1 \geqslant 0 \implies x \geqslant 1$$

2. For the denominator $x-2$ to be non-zero, we need:

$$x-2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x=1$.
(Details omitted as they are not crucial for this case study.)
Since $f(1)=0$ is a real number, our solution is confirmed.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{1}$.

(Details omitted as they are not crucial for this case study.)
1. For $\sqrt{x-1}$ to be defined in the real numbers, we need:

$$x-1 \geqslant 0 \implies x \geqslant 1$$

2. For the denominator $x-2$ to not be zero, we need:

$$x-2 \neq 0 \implies x \neq 2$$

Combining these two conditions, x must be greater than or equal to 1 but not equal to 2.
The smallest integer that satisfies this condition is $x=3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

Figure 14: Case study demonstrating improved exploration: the auxiliary shaping term enables the model to explore and reach the correct solution (continued).

GRPO with Auxiliary Shaping Term#9-#16

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression inside the square root, $x - 1$, must be non-negative because the square root of a negative number is not a real number. Therefore, we need:

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero because division by zero is undefined. Therefore, we need:

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 1$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{1}$.

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression under the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator must not be zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression inside the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

This ensures that $\sqrt{x-1}$ is a real number.
2. **Denominator Condition**: The denominator must not be zero.

$$x - 2 \neq 0 \implies x \neq 2$$

This ensures that the function is defined.
Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. For the square root to be defined, we need:

$$x - 1 \geqslant 0$$

Solving this inequality:

$$x \geqslant 1$$

2. For the denominator to be non-zero, we need:

$$x - 2 \neq 0$$

Solving this equation:

$$x \neq 2$$

Combining these two conditions, x must be greater than or equal to 1 but not equal to 2.
The smallest integer that satisfies this condition is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. The expression inside the square root must be non-negative. 2. The denominator must not be zero.
Let's address these conditions one by one:
1. For the square root to be defined, the expression inside it must be greater than or equal to zero:

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. For the denominator to be non-zero:

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these conditions, we need x to be greater than or equal to 1 but not equal to 2. The smallest integer that satisfies this is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression inside the square root must be non-negative because the square root of a negative number is not a real number.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator must not be zero because division by zero is undefined.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these two conditions, we get:

$$x \geqslant 1 \quad \text{and} \quad x \neq 2$$

The smallest integer that satisfies $x \geqslant 1$ and $x \neq 2$ is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

(Details omitted as they are not crucial for this case study.)
1. **Numerator Condition**: The expression under the square root, $x - 1$, must be non-negative because the square root of a negative number is not a real number.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator, $x - 2$, must not be zero because division by zero is undefined.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these conditions, we need $x \geqslant 1$ and $x \neq 2$. The smallest integer that satisfies these conditions is $x = 1$.
(Details omitted as they are not crucial for this case study.)
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{1}$.

To determine the smallest possible integer value for x such that $f(x) = \frac{\sqrt{x-1}}{x-2}$ has a real number value, we need to ensure that both the numerator and the denominator satisfy their respective conditions for real numbers.
1. **Numerator Condition**: The expression inside the square root must be non-negative.

$$x - 1 \geqslant 0 \implies x \geqslant 1$$

2. **Denominator Condition**: The denominator must not be zero.

$$x - 2 \neq 0 \implies x \neq 2$$

Combining these conditions, we need x to be greater than or equal to 1 but not equal to 2. The smallest integer that satisfies this condition is $x = 3$.
Therefore, the smallest possible integer value for x such that $f(x)$ has a real number value is $\boxed{3}$.

Figure 14: Case study demonstrating improved exploration: the auxiliary shaping term enables the model to explore and reach the correct solution (continued).

Question

Abdul and Chiang are standing 48 feet apart in a field. Bharat is standing in the same field as far from Abdul as possible so that the angle formed by his lines of sight to Abdul and Chiang measures 60°. What is the square of the distance (in feet) between Abdul and Bharat?

Ground Truth

3072

Vanilla GRPO#1-#8

(Details omitted as they are not crucial for this case study.)
1. **Identify the triangle and its properties:**
(Details omitted as they are not crucial for this case study.)
2. **Use the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
3. **Apply the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
4. **Find the square of the distance:**
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{2304}$.

To solve this problem, we can use the Law of Sines in trigonometry. Let's denote the distance between Abdul and Bharat as x and the distance between Chiang and Bharat as y. We know that the angle between Abdul and Chiang as seen from Bharat is 60°, and the distance between Abdul and Chiang is 48 feet.
Given: - $\angle ABC = 60^\circ$ - $AB = 48$ feet - We need to find x^2
First, we apply the Law of Sines in triangle ABC:
(Details omitted as they are not crucial for this case study.)
Thus, the square of the distance between Abdul and Bharat is $\boxed{9216}$.

To solve this problem, we can use the Law of Sines in trigonometry. Let's denote the distance between Abdul and Bharat as AB, and the distance between Chiang and Bharat as BC. Since Bharat is as far from Abdul as possible to form a 60° angle, triangle ABC is a right triangle with the right angle at Chiang. This means $\angle ACB = 90^\circ$ and $\angle BAC = 60^\circ$.
(Details omitted as they are not crucial for this case study.)
Thus, the square of the distance between Abdul and Bharat is $\boxed{9216}$.

First, we need to find the distance AB using the Law of Sines. In triangle ABC:
(Details omitted as they are not crucial for this case study.)
Now, we need to find the square of the distance AB:
(Details omitted as they are not crucial for this case study.)
Thus, the square of the distance between Abdul and Bharat is $\boxed{1728}$.
To verify the accuracy of this result, we can use Python to perform the calculations.
(Details omitted as they are not crucial for this case study.)

To solve this problem, we can use the Law of Sines and some geometric properties. Let's break it down step by step.
1. **Identify the triangle and its properties:**
(Details omitted as they are not crucial for this case study.)
2. **Apply the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
3. **Calculate the distances:**
(Details omitted as they are not crucial for this case study.)
4. **Find the square of the distance:**
(Details omitted as they are not crucial for this case study.)
So, the square of the distance between Abdul and Bharat is $\boxed{768}$.

(Details omitted as they are not crucial for this case study.)
1. Recognize that the triangle ABC is a right triangle with $\angle BAC = 90^\circ$ and $\angle ABC = 60^\circ$. Therefore, $\angle ACB = 30^\circ$.
(Details omitted as they are not crucial for this case study.)
3. Since $AC = 48$ feet, and in a 30-60-90 triangle, the side opposite the 60° angle is $\sqrt{3}$ times the side opposite the 30° angle, we can find AB:

$$AB = 48\sqrt{3}$$

(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{6912}$ feet.

To solve this problem, we need to use some geometric principles, specifically involving the Law of Sines and properties of triangles.
1. **Identify the triangle and the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Use the Law of Sines to find the distance BC (the distance from Bharat to Chiang):**
(Details omitted as they are not crucial for this case study.)
3. **Find the distance AC (the distance from Abdul to Bharat):**
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{11520}$.

(Details omitted as they are not crucial for this case study.)
1. **Identify the triangle and the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Use the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
3. **Maximize AB:**
(Details omitted as they are not crucial for this case study.)
4. **Apply the Pythagorean theorem:**
(Details omitted as they are not crucial for this case study.)
Thus, the square of the distance between Abdul and Bharat is $\boxed{9216}$.

Figure 15: Case study: Geometry problem. GRPO with auxiliary shaping term not only achieves higher accuracy but also explores more diverse reasoning strategies than vanilla GRPO.

Vanilla GRPO#9-#16

To solve this problem, we can use the Law of Sines and some geometric properties. Let's break it down step by step.
1. **Identify the triangle and the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Determine the largest possible angle at A:**
(Details omitted as they are not crucial for this case study.)
3. **Use the Pythagorean theorem:**
(Details omitted as they are not crucial for this case study.)
4. **Find the square of the distance between Abdul and Bharat:**
(Details omitted as they are not crucial for this case study.)
So, the square of the distance between Abdul and Bharat is $\boxed{9216}$.

To solve this problem, we can use some geometric properties and trigonometry. Let's break it down step by step:
1. **Identify the triangle formed**:
(Details omitted as they are not crucial for this case study.)
2. **Use the Law of Sines**:
(Details omitted as they are not crucial for this case study.)
3. **Maximize the distance**:
(Details omitted as they are not crucial for this case study.)
4. **Square the distance**:
(Details omitted as they are not crucial for this case study.)
Let's calculate it:

$$48^2 = 2304$$

So, the square of the distance between Abdul and Bharat is $\boxed{2304}$.

To solve the problem, we need to determine the distance between Abdul and Bharat, given that the angle between their lines of sight to Chiang is 60° and that Bharat is positioned to be as far from Abdul as possible.
(Details omitted as they are not crucial for this case study.)
Using the Law of Cosines in triangle ABC:
(Details omitted as they are not crucial for this case study.)

$$BC^2 = d^2 + 2304 - 48d$$

(Details omitted as they are not crucial for this case study.)
The square of the distance between Abdul and Bharat is:

$$d^2 = 96^2 = 9216$$

Thus, the square of the distance between Abdul and Bharat is $\boxed{9216}$.

(Details omitted as they are not crucial for this case study.)
1. **Identify the known values:**
(Details omitted as they are not crucial for this case study.)
2. **Set up the triangle:**
(Details omitted as they are not crucial for this case study.)
3. **Apply the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
4. **Determine the angles:**
(Details omitted as they are not crucial for this case study.)
5. **Simplify the Law of Sines equation:**
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{2304}$.

To solve this problem, we need to use some geometric properties and trigonometry. Let's break it down step by step.
1. **Understanding the Problem:**
(Details omitted as they are not crucial for this case study.)
2. **Geometric Insight:**
(Details omitted as they are not crucial for this case study.)
3. **Using Trigonometry:**
(Details omitted as they are not crucial for this case study.)
4. **Calculating the Distance:**
- Let the distance AB be x. Since ∠ABC is 60° and we need to maximize AB, we use the Law of Cosines in triangle ABC:
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{432}$.

(Details omitted as they are not crucial for this case study.)
1. **Identify the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Visualize the problem:**
(Details omitted as they are not crucial for this case study.)
3. **Use the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
4. **Properties of an equilateral triangle:**
(Details omitted as they are not crucial for this case study.)
5. **Calculate the square of the distance:**
(Details omitted as they are not crucial for this case study.)
Thus, the square of the distance between Abdul and Bharat is $\boxed{2304}$.
Providethefinalanswerwithin$\boxed{}$.

To solve this problem, we need to use some geometric properties and trigonometry. Let's break it down step by step.
1. **Understand the Problem:**
(Details omitted as they are not crucial for this case study.)
2. **Visualize the Problem:**
(Details omitted as they are not crucial for this case study.)
3. **Use Geometric Properties:**
(Details omitted as they are not crucial for this case study.)
4. **Apply Trigonometry:**
(Details omitted as they are not crucial for this case study.)
5. **Calculate the Distance:** - AB = 2 * AC = 2 * 48 = 96 feet. - The square of the distance AB is 96^2.
6. **Final Calculation:** - $96^2 =$ 9216.
So, the square of the distance between Abdul and Bharat is $\boxed{9216}$.

To solve this problem, we can use the Law of Sines in trigonometry. Let's break it down step by step:
1. **Identify the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Set up the triangle:**
(Details omitted as they are not crucial for this case study.)
3. **Use the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
4. **Find the square of the distance:** The distance between Abdul and Bharat is 24 feet. The square of this distance is:

$$24^2 = 576$$

Therefore, the square of the distance between Abdul and Bharat is $\boxed{576}$.

Figure 15: Case study: Geometry problem. GRPO with auxiliary shaping term not only achieves higher accuracy but also explores more diverse reasoning strategies than vanilla GRPO (continued).

GRPO with Auxiliary Shaping Term#1-#8

(Details omitted as they are not crucial for this case study.)
Given that $\angle ABC = 60^\circ$ and $AC = 48$ feet, we can use the Law of Sines to find AB:
(Details omitted as they are not crucial for this case study.)
The square of the distance AB is:

$$(32\sqrt{3})^2 = 32^2 \times 3 = 1024 \times 3 =$$

Thus, the square of the distance between Abdul and Bharat is $\boxed{3072}$.

To solve this problem, we can use the Law of Sines in trigonometry.
(Details omitted as they are not crucial for this case study.)
Given that $AC = 48$ feet, we can use the Law of Sines to find AB:

$$\frac{AB}{\sin(60^\circ)} = \frac{AC}{\sin(30^\circ)}$$

(Details omitted as they are not crucial for this case study.)
Solving for AB:

$$AB = 96 \times \frac{\sqrt{3}}{2} = 48\sqrt{3}$$

(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{6912}$.

(Details omitted as they are not crucial for this case study.)
1. **Identify the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Visualize the problem:**
(Details omitted as they are not crucial for this case study.)
3. **Determine the type of triangle:** - Since angle ABC and angle ACB are both 60°, triangle ABC is an equilateral triangle. In an equilateral triangle, all sides are equal.
4. **Calculate the side length of the equilateral triangle:** - Since AC = 48 feet and triangle ABC is equilateral, AB (the distance from Abdul to Bharat) is also 48 feet.
(Details omitted as they are not crucial for this case study.)
So, the square of the distance between Abdul and Bharat is $\boxed{2304}$.

To solve this problem, we can use the Law of Sines and some geometric properties. Let's denote the positions of Abdul, Chiang, and Bharat as points A, C, and B respectively. The distance between A and C is given as 48 feet. The angle $\angle ACB$ is 60°. We need to find the square of the distance AB, which is the farthest distance Bharat can be from Abdul.
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{6912}$.

To solve this problem, we need to use some geometric principles. Let's denote the positions of Abdul, Chiang, and Bharat as points A, C, and B respectively. We are given that the distance between A and C (AC) is 48 feet, and the angle ABC is 60°. We need to find the distance between A and B (AB), and then square it.
(Details omitted as they are not crucial for this case study.)
Step 3: Recognize that if triangle ABC is equilateral, then all sides are equal. Therefore, AB = BC = AC = 48 feet.
Step 4: Calculate the square of the distance AB. Since AB = 48 feet, we square this value to get AB².

$$AB^2 = 48^2 = 2304$$

So, the square of the distance between Abdul and Bharat is $\boxed{2304}$.

To find the square of the distance between Abdul and Bharat, we can use the Law of Sines in trigonometry. Let's denote the distance between Abdul and Bharat as AB, and the distance between Chiang and Bharat as BC. Since Bharat is as far from Abdul as possible, triangle ABC is a right triangle with $\angle ACB = 90^\circ$.
Given: - $\angle ABC = 60^\circ$ - $AC = 48$ feet
We want to find AB^2.
First, we apply the Law of Sines in triangle ABC:
(Details omitted as they are not crucial for this case study.)
Thus, the square of the distance between Abdul and Bharat is $\boxed{3072}$.

(Details omitted as they are not crucial for this case study.)
In a right triangle, if one of the angles is 60°, the triangle is a 30-60-90 triangle. In a 30-60-90 triangle, the sides are in the ratio $1 : \sqrt{3} : 2$. Here, AC is the hypotenuse, so we can set up the relationship as follows:

$$AC = 2 \times AB$$

$$48 = 2 \times AB$$

$$AB = \frac{48}{2} = 24 \text{ feet}$$

We need to find the square of the distance AB:

$$AB^2 = 24^2 = 576$$

Thus, the square of the distance between Abdul and Bharat is $\boxed{576}$.

(Details omitted as they are not crucial for this case study.)
Using the properties of a 30-60-90 triangle, we know that the side opposite the 60° angle is $\sqrt{3}$ times the side opposite the 30° angle. Let's denote the side opposite the 30° angle (which is half the distance between Abdul and Chiang) as x. Therefore, $AB = x\sqrt{3}$ and $BC = 2x$.
(Details omitted as they are not crucial for this case study.)

$$AB^2 = 2304(21 - 12\sqrt{3})$$

Since we are asked to find the square of the distance, and we have already calculated it, we can box the final answer:

$$\boxed{2304}$$

Figure 15: Case study: Geometry problem. GRPO with auxiliary shaping term not only achieves higher accuracy but also explores more diverse reasoning strategies than vanilla GRPO (continued).

GRPO with Auxiliary Shaping Term#9-#16

To solve this problem, we can use the Law of Sines in trigonometry.
(Details omitted as they are not crucial for this case study.)
Using the Law of Sines:

$$\frac{AB}{\sin(120^\circ)} = \frac{AC}{\sin(60^\circ)}$$

(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{3072}$.

To solve this problem, we need to use some geometric properties and trigonometric relationships. Let's break it down step by step.
1. **Identify the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Visualize the problem:**
(Details omitted as they are not crucial for this case study.)
3. **Use the Law of Cosines:**
(Details omitted as they are not crucial for this case study.)
4. **Apply the Law of Cosines:**
(Details omitted as they are not crucial for this case study.)
5. **Find the square of the distance:** - The square of the distance between Abdul and Bharat is $d^2 = 2304$.
Therefore, the square of the distance between Abdul and Bharat is $\boxed{2304}$.

To solve this problem, we can use the Law of Sines and some geometric properties.
(Details omitted as they are not crucial for this case study.)
Given that the angle ABC is 60° and AB is maximized, the triangle ABC is a right triangle with angle ACB being 90° (since the angle at Bharat is 60° and the sum of angles in a triangle is 180°, the remaining angle at A must be 30°, making AB the hypotenuse of a 30-60-90 triangle).
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{3072}$.

To solve this problem, we need to use some geometric principles. Let's denote the positions of Abdul, Chiang, and Bharat as points A, C, and B respectively. The distance between A and C is 48 feet, and we need to find the distance AB when the angle $\angle ABC$ is 60° and B is as far from A as possible.
1. **Identify the configuration**:
(Details omitted as they are not crucial for this case study.)
2. **Use the Law of Cosines**:
(Details omitted as they are not crucial for this case study.)
3. **Substitute into the Law of Cosines**:
(Details omitted as they are not crucial for this case study.)
So, the square of the distance between Abdul and Bharat is $\boxed{3840}$ square feet.

To solve this problem, we need to use some geometric principles, specifically involving the Law of Sines and properties of triangles. Let's break it down step by step:
1. **Identify the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Visualize the problem:**
(Details omitted as they are not crucial for this case study.)
3. **Use the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
4. **Determine the angles:**
(Details omitted as they are not crucial for this case study.)
5. **Calculate the distance AB:**
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{3072}$.

To find the square of the distance between Abdul and Bharat, let's denote the distance between Abdul and Bharat as d, and the distance between Chiang and Bharat as d'. Since the angle formed by the lines of sight to Abdul and Chiang is 60°, we can use the Law of Cosines in the triangle formed by Abdul, Chiang, and Bharat. The Law of Cosines states that for any triangle with sides a, b, and c, and the angle θ opposite side c:

$$c^2 = a^2 + b^2 - 2ab\cos(\theta)$$

(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{2304}$.

To solve this problem, we'll use some geometric principles, particularly the Law of Sines and properties of triangles. Let's break down the problem step-by-step:
1. **Identify the given information:**
(Details omitted as they are not crucial for this case study.)
2. **Visualize the problem:**
(Details omitted as they are not crucial for this case study.)
3. **Use the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
4. **Apply the Law of Sines:**
(Details omitted as they are not crucial for this case study.)
5. **Use trigonometry in the right triangle:**
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{1728}$.

To solve this problem, we need to use some geometric properties and trigonometry. Let's denote the positions of Abdul, Chiang, and Bharat as points A, C, and B respectively in the field. We know that AC = 48 feet and the angle ∠ACB = 60°. We want to find the distance AB, which we'll denote as d, and then find d^2.
1. **Identify the type of triangle:**
(Details omitted as they are not crucial for this case study.)
2. **Apply the Law of Cosines:**
(Details omitted as they are not crucial for this case study.)
3. **Maximize the distance d:**
(Details omitted as they are not crucial for this case study.)
Therefore, the square of the distance between Abdul and Bharat is $\boxed{1728}$.

Figure 15: Case study: Geometry problem. GRPO with auxiliary shaping term not only achieves higher accuracy but also explores more diverse reasoning strategies than vanilla GRPO (continued).