

Silence the Judge: Reinforcement Learning with Self-Verifier via Latent Geometric Clustering

Anonymous ACL submission

Abstract

Group Relative Policy Optimization (GRPO) significantly enhances the reasoning performance of Large Language Models (LLMs). However, this success heavily relies on expensive external verifiers or human rules. Such dependency not only leads to significant computational costs and training latency, but also yields sparse rewards that hinder optimization efficiency. To address these challenges, we propose **Latent-GRPO**, a framework that derives intrinsic rewards directly from latent space geometry. Crucially, our empirical analysis reveals a compelling geometric property: terminal token representations of correct reasoning trajectories form dense clusters with high intra-class similarity, whereas incorrect trajectories remain scattered as outliers. In light of this discovery, we introduce the **Iterative Robust Centroid Estimation (IRCE)** algorithm, which generates dense, continuous rewards by mitigating magnitude fluctuations via spherical projection and estimating a robust “truth centroid” through iterative aggregation. Experimental results on multiple datasets show that our method maintains model performance while achieving a training speedup of over $2\times$ compared to baselines. Furthermore, extensive results demonstrate strong generalization ability and robustness.

1 Introduction

Large Language Models (LLMs) (Zhao et al., 2023) have achieved remarkable success in tackling complex reasoning tasks (Hendrycks et al., 2021a; Cobbe et al., 2021; Chen, 2021). To further enhance these capabilities, Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) has been established as the standard paradigm for model alignment. Specifically, this process conventionally utilized Proximal Policy Optimization (PPO) (Rafailov et al., 2023) to refine policy performance. Furthermore, Group Relative

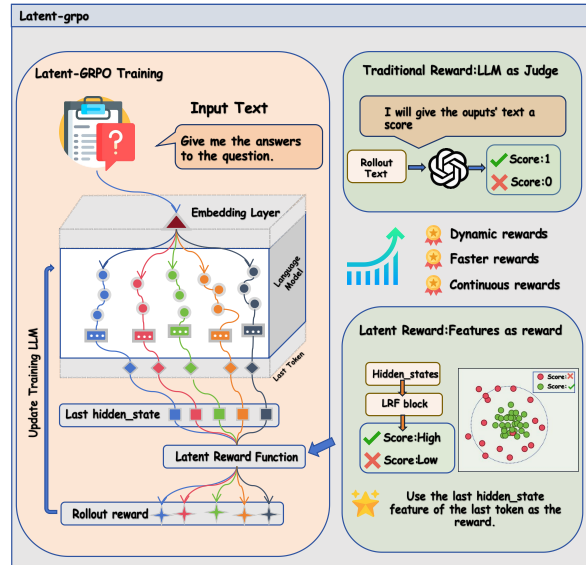


Figure 1: Comparison between conventional GRPO and Latent-GRPO. Conventional GRPO relies on expensive external verifiers to compute rewards, whereas Latent-GRPO autonomously extracts reward signals from the geometric structure of the latent space, eliminating external dependencies.

Policy Optimization (GRPO) (Shao et al., 2024) simplifies the process by replacing the value model with group-based advantages to reduce computational costs.

However, the practical efficacy of GRPO is often constrained by its heavy reliance on external verifiers (Wen et al., 2025; Zheng et al., 2023; Zhou et al., 2025), which makes training outcomes highly sensitive to their quality. On the one hand, rule-based verifiers are typically confined to deterministic tasks like mathematics. Moreover, designing clear and error-free rules for complex reasoning remains extremely difficult, and imperfect rules can severely degrade training performance. On the other hand, employing external LLMs or training additional reward models incurs substantial costs (Lightman et al., 2023). These approaches intro-

duce significant computational overhead and inference latency, which ultimately slows down the entire training process. Furthermore, such external judges are susceptible to biases or inaccurate scoring, thereby compromising training stability and final model quality (Xu et al., 2025; Cai et al., 2025). Beyond these concerns, most existing reward signals remain sparse and discrete (Tao et al., 2025). This binary feedback fails to capture the continuous semantic nuances of the reasoning process, often leading the model toward reward hacking (Cui et al., 2025; Gao et al., 2023). To address these challenges, we argue that an ideal reward mechanism should be **intrinsic, dense, and training-free**.

Building on representation engineering (Zou et al., 2023; Bartoszcze et al., 2025), which reveals that LLMs encode high-level semantic concepts internally (Marks and Tegmark, 2023), we discover a striking geometric property: the **last hidden states of terminal tokens** in correct reasoning trajectories form dense clusters, while incorrect paths remain scattered. Theoretically, this stems from the Transformer’s attention mechanism, which progressively aggregates the reasoning context into the final representation. Furthermore, this geometric consistency reflects the model’s inherent discriminative capabilities acquired during large-scale pre-training (Radford et al., 2018). In essence, the latent space acts as an implicit verifier where logical consistency manifests as semantic convergence, providing a robust foundation for intrinsic reward modeling.

We introduce **Latent-GRPO**, a framework that utilizes robust intrinsic rewards from the geometric properties of the latent space, as shown in Figure 1. At its core, the **Iterative Robust Centroid Estimation (IRCE)** algorithm identifies a “truth centroid” from the **last hidden states** of terminal tokens, using their geometric relationship as a continuous reward. Unlike rule-based verifiers, our approach yields dense reward signals, which allows for more granular optimization across a wider range of reasoning scenarios where clear rules are unavailable. Compared to LLM-as-judge, Latent-GRPO eliminates external model dependencies, which significantly reduces training latency and prevents model collapse caused by inconsistent or noisy judging. More importantly, this approach effectively activates the rich reasoning knowledge acquired by the model during its large-scale pre-training phase. Detailed experiments on GSM8K, MATH, and Open-Platypus all show that Latent-

GRPO achieves 2× training speedup compared to LLM-as-Judge across three model scales (0.6B, 1.7B, and 4B). While it exceeds accuracy of both LLM-as-Judge and Rule-based Methods. Furthermore, additional analyses underscore the robustness and generalization ability of our approach across diverse scenarios.

2 Related Work

Policy Optimization and Group-based Variants.

Recent advances in reinforcement learning for LLMs have focused on improving training efficiency and stability. PPO (Schulman et al., 2017) was the foundational algorithm for RLHF (Ouyang et al., 2022; Ziegler et al., 2019; Stiennon et al., 2020), but requires maintaining a critic model with significant memory overhead. DPO (Rafailov et al., 2023) eliminated the critic but limits exploration capabilities. GRPO (Shao et al., 2024) introduced group-based advantage estimation to balance online exploration and computational efficiency. Recent variants (Yu et al., 2025; Zheng et al., 2025; Liu et al., 2025; Zhao et al., 2025) address specific challenges at the optimizer level.

Training-free evaluation Methods. To reduce dependence on expensive external supervision, researchers have explored training-free evaluation approaches. Methods like Self-Consistency (Wang et al., 2022), Self-Refine (Madaan et al., 2023), Tree-of-Thoughts (Yao et al., 2023), Best-of-N (Stiennon et al., 2020), and Forest-of-Thoughts (Bi et al., 2024) leverage consensus mechanisms to identify high-quality outputs.

Latent Space and Latent Thinking. The latent space of LLMs is known to encode rich semantic information (Zhang et al., 2023; Goyal et al., 2023; Hao et al., 2024; Geiping et al., 2025). Recent work has explored latent thinking, using hidden states for self-evaluation and reward prediction. Methods like CoE (Wang et al., 2024), LTO (Du et al., 2025), LaTRO (Chen et al., 2024), and EndoRM (Li et al., 2025) leverage latent representations for RL guidance. *For comprehensive related work discussion, please refer to Appendix B.*

3 Geometric Properties of Latent Space

This section demonstrates that the LLM latent space intrinsically captures reasoning quality through a series of empirical analyses. Specifically, we observe that correct reasoning trajectories exhibit high geometric clustering in their termi-

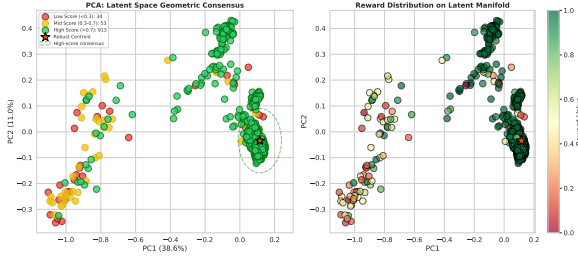


Figure 2: 2D PCA projection of 1,000 rollouts. Correct trajectories (green) form a dense consensus core around the truth centroid (gold star), while incorrect ones (red) scatter as outliers.

nal hidden states, whereas incorrect paths remain scattered. Building on this, we explore leveraging these geometric features to score reasoning quality, ultimately finding a high degree of consistency with evaluations from external model-based verifiers. This alignment confirms the latent space as a robust and autonomous source of reward signals.

3.1 Theoretical Motivation

Our analysis is guided by two fundamental properties of the Transformer architecture. First, the last hidden state of the terminal token, h_T , acts as a semantic summary of the entire reasoning chain (Afzal et al., 2025; Wang et al., 2024). Specifically, as the final representation before the language model head, h_T effectively aggregates the model’s converged reasoning information (Shai et al., 2024). Second, successful reasoning often leads to semantic collapse (Papayan et al., 2020; Wang et al., 2022). This means that while intermediate steps may vary, all correct trajectories tend to converge toward a unified semantic endpoint. In contrast, incorrect paths typically scatter due to their diverse failure modes (Marks and Tegmark, 2023; Zou et al., 2023).

3.2 Analysis Design

To validate our findings, we conduct two analyses on the GSM8K dataset. First, we generate 1,000 independent trajectories per prompt to examine global clustering patterns. Second, we construct GRPO groups ($G = 8$) to evaluate the effectiveness of geometric reward assignment. In both cases, we extract the last hidden state of the terminal token, $h_T \in \mathbb{R}^{1024}$, and compute distances directly in the original high-dimensional space to preserve the full semantic information. To verify these geometric rewards, we use GPT-4o to obtain ground-truth labels; critically, these labels are used only for val-

idation in this section and are not required by our training framework.

3.3 Large-Scale Clustering Analysis

Figure 2 visualizes the **last hidden states** of terminal tokens via PCA, revealing a clear “core-periphery” structure. Analyzing 1,000 trajectories from GSM8K, we observe two key patterns: (1) **High-Density Consensus Core**, where 913 correct trajectories cluster tightly around the centroid ($d_{\text{correct}} = 0.249$); and (2) **Dispersed Outlier Region**, where 34 incorrect trajectories remain scattered ($d_{\text{incorrect}} = 1.029$). The resulting $4.13 \times$ distance ratio confirms significant geometric separability of reasoning quality based on these **last hidden states**. To demonstrate the universality of these patterns, we provide additional observations across model scales (0.6B to 4B) and diverse datasets (e.g., ScienceQA and ARB) in Appendix I.1.

3.4 Validation via GRPO Group Simulation

Within each GRPO group, we calculate a robust centroid \mathbf{c} via weighted aggregation of the terminal tokens’ **last hidden states**. The latent geometric score is then defined as $s_{\text{latent}}(i) = -\|\mathbf{h}_i - \mathbf{c}\|_2$. Figure 4 validates this approach: (a) presents a box plot of the reward distribution across different quality levels. It reveals that incorrect trajectories exhibit high dispersion and a lower median, whereas correct trajectories are significantly more concentrated with much higher reward values. (b-d) demonstrate high ranking consistency in representative groups. Notably, our method achieves a maximum Spearman correlation of $\rho = 0.927$ and a Top-1 selection agreement of up to 85% with external verifiers. These findings confirm that latent geometric properties can serve as a robust, training-free alternative to external judges, providing a solid foundation for Latent-GRPO. For a more detailed analysis, please refer to Appendix C.

Altogether, these analysis demonstrate that the last hidden states of terminal tokens inherently capture reasoning quality through their geometric structure. This latent consistency provides a strong foundation for designing efficient reward mechanisms independent of external supervision. Building on these insights, we introduce the Latent-GRPO framework, which transforms these geometric properties into dense, continuous reward signals for policy optimization.

Framework of Latent-GRPO

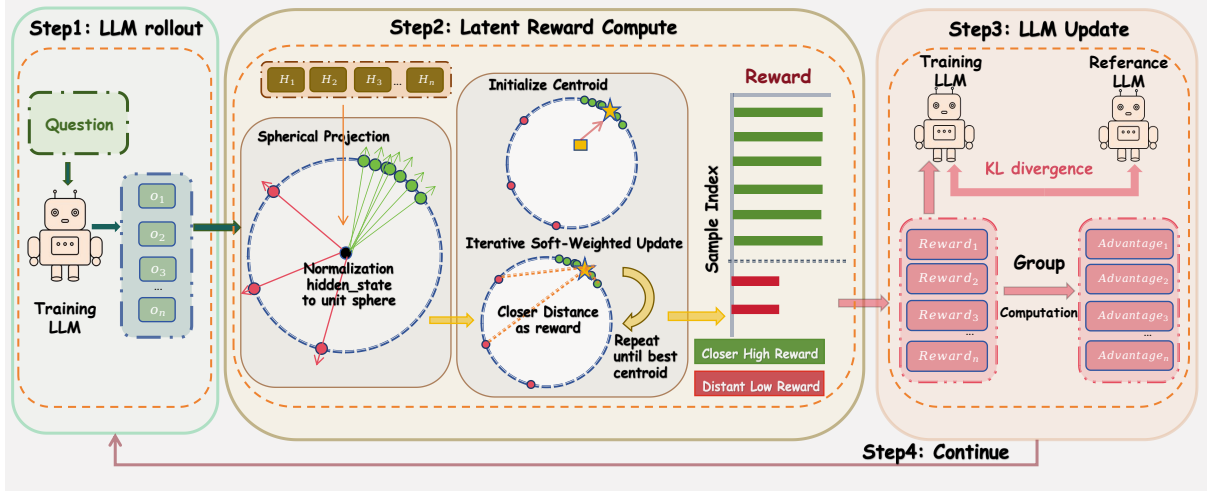


Figure 3: Overview of the Latent-GRPO framework. The policy model generates a group of responses for each prompt. Instead of relying on external verifiers, we extract the hidden states of the last token from each trajectory and apply the Iterative Robust Centroid Estimation (IRCE) algorithm to compute intrinsic rewards based on geometric clustering in the latent space. These rewards are then used to compute group-relative advantages for policy optimization. The entire process operates within the latent space, achieving zero additional inference overhead while providing dense reward signals.

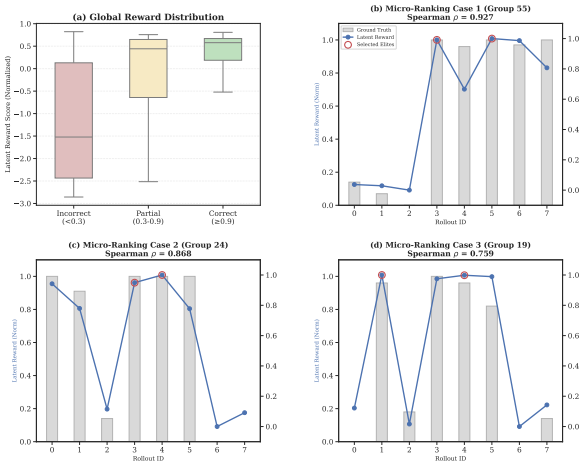


Figure 4: Validation of geometric-based scoring. (a) Distribution separability across quality levels (correct, partial, incorrect). (b-d) Group-level ranking consistency in representative 8-trajectory groups.

4 Methodology: Latent-GRPO

4.1 GRPO Short Review

Group Relative Policy Optimization (GRPO) (Shao et al., 2024) optimizes policy performance by computing relative advantages across a group of G trajectories $\{y_1, \dots, y_G\}$ sampled from the same prompt x :

$$A_i = \frac{R_i - \bar{R}_{\text{group}}}{\text{std}(R_{\text{group}}) + \epsilon} \quad (1)$$

where R_i represents the reward for the i -th trajectory. While architecturally efficient, standard GRPO faces two major challenges: heavy reliance on external verifiers, which leads to high costs and latency, and reward sparsity, which offers limited guidance for complex reasoning tasks. To address these issues, we propose a reward mechanism characterized by two key properties: (1) **intrinsic**, utilizing the model’s own **last hidden states of terminal tokens**, and (2) **dense**, providing continuous quality scores that enhance optimization gradients.

4.2 Core Algorithm: Iterative Robust Centroid Estimation

Based on the geometric findings in Section 3, we introduce the Iterative Robust Centroid Estimation (IRCE) algorithm to extract high-fidelity reward signals. For each prompt, the policy model generates a group of G trajectories. We first extract the **last hidden states of terminal tokens** $\mathbf{h}_i \in \mathbb{R}^d$ ($i = 1, \dots, G$) and perform spherical normalization:

$$\tilde{\mathbf{h}}_i = \frac{\mathbf{h}_i}{\|\mathbf{h}_i\|_2} \quad (2)$$

This operation projects all trajectories onto a unit hypersphere, effectively eliminating magnitude fluctuations while ensuring the subsequent analysis focuses exclusively on semantic directionality.

The core of IRCE is to dynamically estimate a consensus centroid μ that represents the “truth direction” of reasoning within the group. To suppress the influence of erroneous outliers, we employ an iterative soft-weighting mechanism. In each iteration s , we compute soft weights $w_i^{(s)}$ using a Gaussian kernel based on the sample’s current distance to the centroid:

$$w_i^{(s)} = \frac{\exp\left(-\frac{(d_i^{(s)})^2}{2(\sigma^{(s)})^2}\right)}{\sum_{j=1}^G \exp\left(-\frac{(d_j^{(s)})^2}{2(\sigma^{(s)})^2}\right)} \quad (3)$$

where $\sigma^{(s)}$ is an adaptive scale parameter derived from the group’s distance distribution. The centroid is subsequently updated via weighted aggregation: $\mu^{(s+1)} = \text{Norm}(\sum w_i^{(s)} \tilde{\mathbf{h}}_i)$. **Upon convergence** after T iterations, the intrinsic reward R_i is defined as the negative Euclidean distance to the final centroid:

$$R_i = \frac{-\|\tilde{\mathbf{h}}_i - \mu^{(T)}\|_2 - \min_j(-\|\tilde{\mathbf{h}}_j - \mu^{(T)}\|_2)}{\max_j(-\|\tilde{\mathbf{h}}_j - \mu^{(T)}\|_2) - \min_j(-\|\tilde{\mathbf{h}}_j - \mu^{(T)}\|_2)} \quad (4)$$

This mechanism yields intrinsic rewards that are **dense, continuous, and naturally bounded**. By applying Min-Max normalization, we map the distances into a calibrated range $R_i \in [0, 1]$, which prevents gradient explosion and ensures optimization stability across different reasoning tasks. The complete procedure, including the adaptive scaling and normalization, is summarized in Algorithm 1. Furthermore, we provide a rigorous mathematical derivation of this iterative weighting scheme in Appendix D to demonstrate its robustness against sampling noise.

4.3 Latent-GRPO Framework

As illustrated in Figure 3, **Latent-GRPO** integrates the IRCE algorithm into the GRPO pipeline. Unlike traditional reinforcement learning which relies on static external verifiers, Latent-GRPO adopts a dynamic, intrinsic reward mechanism that operates without external supervision.

Dynamic Adaptability. As the policy model evolves during training, its latent representation space naturally shifts. Unlike static external verifiers, IRCE dynamically estimates the consensus centroid for each training batch, allowing reward signals to adapt in real-time to the model’s current state. This online adaptation effectively mitigates distribution shift—a common challenge when reward signals become misaligned with evolving

Algorithm 1 Iterative Robust Centroid Estimation

Input: Hidden states $\{\mathbf{h}_1, \dots, \mathbf{h}_G\}$, max iterations T
Output: Rewards $\{R_1, \dots, R_G\}$
1: // **Step 1: Spherical Projection**
2: **for** $i = 1$ to G **do**
3: $\tilde{\mathbf{h}}_i \leftarrow \mathbf{h}_i / \|\mathbf{h}_i\|_2$
4: **end for**
5: // **Step 2: Initialize Centroid**
6: $\mu \leftarrow \frac{1}{G} \sum_{i=1}^G \tilde{\mathbf{h}}_i$
7: $\mu \leftarrow \mu / \|\mu\|_2$
8: // **Step 3: Iterative Soft-Weighted Update**
9: **for** $s = 0$ to $T - 1$ **do**
10: **for** $i = 1$ to G **do**
11: $d_i \leftarrow \|\tilde{\mathbf{h}}_i - \mu\|_2$
12: **end for**
13: $\sigma \leftarrow \text{std}(\{d_1, \dots, d_G\}) + \epsilon$
14: **for** $i = 1$ to G **do**
15: $w_i \leftarrow \exp(-(d_i)^2 / (2\sigma^2))$
16: **end for**
17: $\mathbf{w} \leftarrow \mathbf{w} / \sum_j w_j$ ▷ Normalize weights
18: $\mu \leftarrow \sum_i w_i \tilde{\mathbf{h}}_i$
19: $\mu \leftarrow \mu / \|\mu\|_2$ ▷ Normalize centroid
20: **end for**
21: // **Step 4: Compute Rewards**
22: **for** $i = 1$ to G **do**
23: $d_i \leftarrow \|\tilde{\mathbf{h}}_i - \mu\|_2$
24: $R_i \leftarrow -d_i$
25: **end for**
26: $\mathbf{R} \leftarrow \text{MinMaxNormalize}(\mathbf{R})$
27: **return** \mathbf{R}

model representations. Additionally, the iterative soft-weighting mechanism ensures stable and robust gradient signals even when initial sample quality is suboptimal.

Computational Efficiency. Latent-GRPO eliminates the computational overhead of external verification by leveraging hidden states already computed during rollout. While conventional verifiers require additional forward passes scaling as $O(GL)$ (where G is group size and L is sequence length), IRCE operates on the last hidden states of terminal tokens with complexity $O(GTd)$ (where T is the number of iterations and d is the latent dimension). Since d is fixed by model architecture and $T \ll L$, the overhead becomes negligible. By eliminating the need for separate reward models or value functions, Latent-GRPO simultaneously reduces memory footprint and transforms reward computation from an external bottleneck into an efficient intrinsic process.

5 Experimental Setup

This section describes our experimental configuration, including dataset selection, evaluation metrics, hardware setup, and the design of our main experiment and ablation studies.

Dataset	Method	Qwen3-0.6B		Qwen3-1.7B		Qwen3-4B	
		Acc ↑	Time ↓	Acc ↑	Time ↓	Acc ↑	Time ↓
<i>GSM8K</i>	LLM-as-Judge	53.52%	768.42m	64.20%	1032.55m	72.12%	1411.72m
	Rule-based	58.41%	434.61m	71.55%	488.63m	79.87%	651.45m
	Latent-GRPO (Ours)	61.25%	431.18m	73.88%	492.34m	82.34%	658.21m
<i>MATH</i>	LLM-as-Judge	52.94%	1224.15m	65.77%	1608.34m	77.44%	2357.31m
	Rule-based	55.63%	723.12m	42.14%	814.22m	62.63%	1084.72m
	Latent-GRPO (Ours)	58.47%	718.63m	78.51%	811.51m	77.53%	1081.47m
<i>Open-Platypus</i>	LLM-as-Judge	34.45%	1937.82m	56.69%	2573.41m	65.21%	3522.18m
	Latent-GRPO (Ours)	40.56%	1079.27m	64.82%	1218.92m	78.06%	1632.52m

Table 1: Comprehensive comparison of reward methods across datasets and model scales. The results demonstrate that Latent-GRPO consistently achieves superior accuracy and training efficiency (Time per epoch) compared to LLM-as-Judge and Rule-based baselines. (Simulated QPS for LLM-as-Judge is 2)

5.1 Datasets and Evaluation Metrics

Datasets. We evaluate our approach on three complementary datasets that span different reasoning complexity levels and domains. **GSM8K** (Cobbe et al., 2021) contains elementary-level mathematical word problems, establishing a foundation for basic arithmetic reasoning. **MATH** (Hendrycks et al., 2021a) comprises high school and competition-level problems, testing more sophisticated mathematical reasoning. **Open-Platypus** (Lee et al., 2023a) covers diverse reasoning tasks across physics, logic, and mathematics, enabling validation across multiple domains. Together, these datasets provide comprehensive coverage for evaluating both accuracy improvements and computational efficiency across varying reasoning complexities. The split of train/test datasets and detailed dataset statistics are provided in Appendix E.

Evaluation Metrics. We measure two primary performance dimensions: (1) **Task Accuracy** on each test dataset of benchmark to assess reasoning capability, and (2) **Training Efficiency** measured as time per epoch across different reward methods under identical experimental settings.

5.2 Experimental Setup

Models and Hardware. We validate our method across three Qwen models of varying sizes: Qwen3-0.6B, Qwen3-1.7B, and Qwen3-4B. This range enables assessment of our method’s effectiveness across different model scales. All experiments are conducted on a single GPU. For detailed hyperparameter settings including GRPO training configuration, IRCE algorithm parameters, and hardware specifications, please refer to Appendix F.

Main Experiment We compare three reward paradigms across three datasets (GSM8K, MATH,

Method	Acc ↑	Time (m) ↓
Qwen3-0.6B		
Mean Pooling	58.74%	435.22
Weighted Mean	57.12%	442.89
Last Token (Ours)	61.25%	431.18
Qwen3-1.7B		
Mean Pooling	71.05%	497.61
Weighted Mean	69.88%	512.14
Last Token (Ours)	73.88%	492.34
Qwen3-4B		
Mean Pooling	79.45%	664.33
Weighted Mean	78.12%	685.56
Last Token (Ours)	82.34%	658.21

Table 2: Comparison of different hidden state extraction methods. Using the terminal token’s representation (Last Token) consistently yields the best reasoning performance with the lowest computational latency.

Open-Platypus) and three model sizes (Qwen3-0.6B, Qwen3-1.7B, Qwen3-4B): (1) **LLM-as-Judge** using external verification (GPT-4o), (2) **Rule-based** verification with ground-truth labels, and (3) our proposed **Latent-GRPO** leveraging intrinsic latent space geometry. This comparison evaluates whether intrinsic geometric signals can match or exceed external verification in both accuracy and training efficiency. Detailed baseline descriptions are provided in Appendix G.1.

6 Results and Analysis

This section presents a comprehensive evaluation of **Latent-GRPO** across three dimensions: performance and efficiency compared to external verifiers, core design (IRCE) choices through ablation studies, and generalization across model families and unseen tasks.

6.1 Reward Methods Comparison

We compare three reward paradigms on GSM8K and MATH: LLM-as-Judge (GPT-4o), Rule-based verification, and Latent-GRPO. For Open-Platypus, we focus on LLM-as-Judge and Latent-GRPO since rule-based methods are limited to tasks with deterministic ground truth.

As shown in Label 1, Latent-GRPO achieves approximately **2× training speedup** compared to LLM-as-Judge across all datasets and model scales. Meanwhile, Latent-GRPO maintains or exceeds the accuracy of both LLM-as-Judge and Rule-based verification. Detailed per-dataset and per-model results are provided in Appendix H.1.

Efficiency. Latent-GRPO eliminates the external verifier bottleneck that constrains LLM-as-Judge. The latter faces two system-level costs: API rate limiting (2 QPS) forces scoring requests into a queue, and each API call incurs 1-2 minutes of latency. Together, these consume 58-63% of total training time. Latent-GRPO instead computes rewards on hidden states already available from the forward pass using the IRCE algorithm, requiring only geometric operations with $O(GTd)$ complexity (where $T \ll L$, typically 5 iterations vs. 2048 sequence length), compared to $O(GL)$ for transformer-based verifiers. Rule-based verification operates at similar cost to Latent-GRPO, confirming that speedup comes primarily from eliminating external calls.

Accuracy. This advantage stems from two factors. First, Latent-GRPO provides continuous, dense rewards based on distance to the consensus centroid, whereas Rule-based and LLM-as-Judge provide only binary 0/1 feedback. Richer reward signals enable more effective policy optimization. Second, Latent-GRPO derives rewards from the model’s internal geometry rather than external judges, avoiding inconsistency and noise from external verifiers. The model no longer depends on external verification accuracy, which stabilizes training and prevents collapse.

6.2 Ablation Studies: Core Design Choices

We validate two critical design choices through systematic ablation studies: (1) The hidden state extraction method for capturing reasoning quality, and (2) The centroid estimation algorithm for computing intrinsic rewards. Detailed quantitative analysis and per-model comparisons are provided in Appendix H.2.

Method	Acc ↑	Time (m) ↓
Qwen3-0.6B		
Mean Pool	57.12%	452.34
K-Means	58.85%	489.12
Eigen Centrality	59.43%	468.76
IRCE (Ours)	61.25%	431.18
Qwen3-1.7B		
Mean Pool	68.45%	512.67
K-Means	70.12%	543.89
Eigen Centrality	71.56%	531.42
IRCE (Ours)	73.88%	492.34
Qwen3-4B		
Mean Pool	77.89%	682.12
K-Means	79.23%	725.67
Eigen Centrality	80.56%	708.45
IRCE (Ours)	82.34%	658.21

Table 3: Comparison of different latent consensus scoring methods on GSM8K. IRCE (Ours) achieves the highest reasoning accuracy with the minimum computational overhead across all model scales.

Hidden State Extraction. We compare three approaches for extracting reasoning quality from hidden states: (1) **Last Token**, using only the terminal token’s representation; (2) **Mean Pooling**, averaging representations across all tokens; (3) **Weighted Mean**, computing weighted averages of key tokens (weighted averaging of pre-designed key tokens such as mathematical operators, structural markers, and reasoning keywords).

As illustrated in Table 2, the Last Token method consistently outperforms all aggregation baselines across various model scales. This finding reveals a fundamental property of transformer-based reasoning: the final token, as the immediate precursor to the end-of-sequence (EOS) prediction, acts as a semantic bottleneck where reasoning correctness is crystallized. In contrast, Mean Pooling degrades performance by incorporating noise from intermediate tokens that are often orthogonal to the final correctness. The failure of Weighted Mean, which focuses on linguistically significant keywords (e.g., "therefore", "solution") is particularly instructive. It suggests that reasoning quality is not localized at specific lexical markers but emerges holistically through the generation process, ultimately converging into the final representation.

Centroid Estimation We compare four approaches for estimating the consensus centroid from a group of trajectories: (1) **Mean Pooling**, (2) **K-Means clustering**, (3) **Eigen Centrality**, and (4) **IRCE**, our proposed iterative robust centroid estimation. All experiments are conducted on GSM8K across

Method	AIME24 \uparrow	AIME25 \uparrow	MATH-500 \uparrow	MMLU \uparrow	BBH \uparrow	Avg \uparrow	Time (m) \downarrow
Qwen3-0.6B							
Base	10.7	15.1	77.6	52.8	41.5	39.54	–
GRPO (LLM-Judge)	9.4	11.7	65.9	50.6	42.4	36.00	4082.96
Latent-GRPO (Ours)	10.6	19.2	67.3	49.9	39.2	37.24	2280.52
Qwen3-1.7B							
Base	48.3	36.8	93.4	62.6	54.5	59.12	–
GRPO (LLM-Judge)	48.1	33.8	90.6	67.6	67.4	61.50	4988.84
Latent-GRPO (Ours)	44.6	35.4	91.2	68.1	68.7	61.60	2340.05
Qwen3-4B							
Base	73.8	65.6	97.0	83.7	72.6	78.54	–
GRPO (LLM-Judge)	72.9	63.3	96.4	85.5	81.9	80.00	6753.47
Latent-GRPO (Ours)	74.6	66.7	97.5	88.5	82.3	81.92	3108.44

Table 4: Performance and efficiency comparison across Qwen3 scales on mathematics (AIME, MATH-500) and general reasoning (MMLU, BBH) benchmarks.

Method	Acc \uparrow	Time (m) \downarrow
GSM8K (2k steps)		
LLM-as-Judge	71.34%	1284.56
Latent-GRPO	78.62%	591.24
MATH (3k steps)		
LLM-as-Judge	43.12%	1945.12
Latent-GRPO	52.45%	912.87
Open-Platypus (4.5k steps)		
LLM-as-Judge	61.88%	2915.68
Latent-GRPO	73.12%	1386.42

Table 5: Experimental results of Latent-GRPO on Llama3.2-3B. Following the same protocol as Qwen3 experiments, Latent-GRPO demonstrates superior scaling performance and significantly reduced training latency compared to the LLM-as-judge baseline.

three model scales using the Last Token extraction method identified above. For details of these latent reward methods, please refer to Appendix G.2.

As shown in Table 3, IRCE consistently outperforms all baseline methods across all model scales. Mean Pooling fails to handle outliers effectively, resulting in suboptimal centroids. K-Means attempts to improve through hard cluster assignment and achieves competitive accuracy, yet still underperforms IRCE. Eigen Centrality uses graph-based importance weighting but introduces significant computational overhead through eigendecomposition, achieving lower accuracy while incurring higher computational cost. Our method, IRCE, maintains both robustness and efficiency, making it the optimal choice for latent reward estimation.

6.3 Generalization

Capability Preservation. A critical concern in RL training is the potential loss of general capabilities due to task-specific overfitting. To address

this, we train Latent-GRPO on a specific mixture of reasoning datasets and subsequently evaluate it on a suite of unseen general benchmarks, including MMLU, AIME (24 & 25), BBH, and MATH-500. As illustrated in Table 4, Latent-GRPO consistently maintains or surpasses the performance of base models across these diverse tasks. These results confirm that dense intrinsic rewards guide the model toward mastering transferable reasoning patterns rather than overfitting.

Cross-Model Generalization. To validate that Latent-GRPO generalizes beyond the Qwen3 series, we extend our evaluation to Llama3.2-3B on the GSM8K, MATH, and Open-Platypus datasets. As shown in Table 5, the latent space geometry provides a universal signal for reasoning quality that is independent of the model family.

7 Conclusion

In this work, we introduce Latent-GRPO, a training framework designed to overcome the efficiency bottlenecks and sparse rewards inherent in verifier-dependent reinforcement learning. At the core of this framework lies the Iterative Robust Centroid Estimation algorithm, which transforms latent geometry into dense rewards, eliminating the reliance on slow external verifiers and achieving a substantial acceleration in training compared to the LLM-as-a-judge baseline. Furthermore, extensive evaluations on unseen benchmarks confirm that our approach maintains competitive accuracy. It effectively preserves native capabilities and prevents task-specific overfitting. These findings substantiate that LLMs possess inherent self-evaluation mechanisms and offer a scalable paradigm for verifier-free post-training.

540 Limitations

541 Our work has two primary limitations that we aim
542 to address in future research. First, while **Latent-**
543 **GRPO** is effective up to 8B parameters, its scaling
544 behavior in ultra-large models (70B+) and appli-
545 cability to open-ended generation remain to be ex-
546 plored. Second, while we provide strong empirical
547 evidence, a formal mathematical framework for la-
548 tent clustering is still nascent. Moving forward, we
549 plan to extend the geometric consensus hypothesis
550 to broader tasks and investigate hybrid frameworks
551 that integrate **IRCE** rewards with offline paradigms
552 like DPO to further stabilize self-supervised align-
553 ment.

554 References

555 Anum Afzal, Florian Matthes, Gal Chechik, and Yf-
556 tah Ziser. 2025. Knowing before saying: Lm
557 representations encode information about chain-of-
558 thought success before completion. *arXiv preprint*
559 *arXiv:2505.24362*.

560 Lukasz Bartoszczke, Sarthak Munshi, Bryan Sukidi, Jen-
561 nifer Yen, Zejia Yang, David Williams-King, Linh
562 Le, Kosi Asuzu, and Carsten Maple. 2025. Rep-
563 resentation engineering for large-language models:
564 Survey and research challenges. *arXiv preprint*
565 *arXiv:2502.17601*.

566 Zhenni Bi, Kai Han, Chuanjian Liu, Yehui Tang, and
567 Yunhe Wang. 2024. Forest-of-thought: Scaling test-
568 time compute for enhancing llm reasoning. *arXiv*
569 *preprint arXiv:2412.09078*.

570 Xin-Qiang Cai, Wei Wang, Feng Liu, Tongliang Liu,
571 Gang Niu, and Masashi Sugiyama. 2025. Re-
572 inforcement learning with verifiable yet noisy re-
573 wards under imperfect verifiers. *arXiv preprint*
574 *arXiv:2510.00915*.

575 Haolin Chen, Yihao Feng, Zuxin Liu, Weiran Yao, Ak-
576 shara Prabhakar, Shelby Heinecke, Ricky Ho, Phil
577 Mui, Silvio Savarese, Caiming Xiong, and 1 oth-
578 ers. 2024. Language models are hidden reason-
579 ers: Unlocking latent reasoning capabilities via self-
580 rewarding. *arXiv preprint arXiv:2411.04282*.

581 Mark Chen. 2021. Evaluating large language models
582 trained on code. *arXiv preprint arXiv:2107.03374*.

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

589 Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang,
590 Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang

591 He, Yuchen Fan, Tianyu Yu, and 1 others. 2025. Pro-
592 cess reinforcement through implicit rewards. *arXiv*
593 *preprint arXiv:2502.01456*.

594 Hanwen Du, Yuxin Dong, and Xia Ning. 2025. La-
595 tent thinking optimization: Your latent reasoning lan-
596 guage model secretly encodes reward signals in its
597 latent thoughts. *arXiv preprint arXiv:2509.26314*.

598 Nan Du, Yanping Huang, Andrew M Dai, Simon Tong,
599 Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun,
600 Yanqi Zhou, Adams Wei Yu, Orhan Firat, and 1 oth-
601 ers. 2022. Glam: Efficient scaling of language mod-
602 els with mixture-of-experts. In *International confer-*
603 *ence on machine learning*, pages 5547–5569. PMLR.

604 Leo Gao, John Schulman, and Jacob Hilton. 2023. Scal-
605 ing laws for reward model overoptimization. In *In-*
606 *ternational Conference on Machine Learning*, pages
607 10835–10866. PMLR.

608 Jonas Geiping, Sean McLeish, Neel Jain, John Kirchen-
609 bauer, Siddharth Singh, Brian R Bartoldson, Bhavya
610 Kailkhura, Abhinav Bhatele, and Tom Goldstein.
611 2025. Scaling up test-time compute with latent rea-
612 soning: A recurrent depth approach. *arXiv preprint*
613 *arXiv:2502.05171*.

614 Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Kr-
615 ishna Menon, Sanjiv Kumar, and Vaishnavh Nagara-
616 jan. 2023. Think before you speak: Training lan-
617 guage models with pause tokens. *arXiv preprint*
618 *arXiv:2310.02226*.

619 Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li,
620 Zhiting Hu, Jason Weston, and Yuandong Tian. 2024.
621 Training large language models to reason in a contin-
622 uous latent space. *arXiv preprint arXiv:2412.06769*.

623 Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul
624 Arora, Steven Basart, Eric Tang, Dawn Song, and
625 Jacob Steinhardt. 2021a. Measuring mathemati-
626 cal problem solving with the math dataset. *arXiv*
627 *preprint arXiv:2103.03874*.

628 Dan Hendrycks and 1 others. 2020. Measuring massive
629 multitask language understanding. *arXiv preprint*
630 *arXiv:2009.03300*.

631 Dan Hendrycks and 1 others. 2021b. Measuring math-
632 ematical problem solving with the math dataset.
633 *NeurIPS*.

634 Robert A Jacobs, Michael I Jordan, Steven J Nowlan,
635 and Geoffrey E Hinton. 1991. Adaptive mixtures of
636 local experts. *Neural computation*, 3(1):79–87.

637 Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellemp-
638 pudi, Dipankar Das, Kunal Banerjee, Sasikanth Avan-
639 cha, Dharma Teja Vooturi, Nataraj Jammalamadaka,
640 Jianyu Huang, Hector Yuen, and 1 others. 2019. A
641 study of bfloat16 for deep learning training. *arXiv*
642 *preprint arXiv:1905.12322*.

643 Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. 2023a.
644 Platypus: Quick, cheap, and powerful refinement of
645 llms. *arXiv preprint arXiv:2308.07317*.

646	Harrison Lee, Samrat Phatale, Hassan Mansoor, Kel-	<i>Advances in Neural Information Processing Systems</i> ,	701
647	lie Ren Lu, Thomas Mesnard, Johan Ferret, Colton	37:75012–75034.	702
648	Bishop, Ethan Hall, Victor Carbune, and Abhinav		
649	Rastogi. 2023b. Rlaif: Scaling reinforcement learn-	Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu,	703
650	ing from human feedback with ai feedback.	Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan	704
		Zhang, YK Li, Yang Wu, and 1 others. 2024.	705
651	Yi-Chen Li, Tian Xu, Yang Yu, Xuqin Zhang, Xiong-	Deepseekmath: Pushing the limits of mathematical	706
652	Hui Chen, Zhongxiang Ling, Ningjing Chao, Lei	reasoning in open language models. <i>arXiv preprint</i>	707
653	Yuan, and Zhi-Hua Zhou. 2025. Generalist reward	<i>arXiv:2402.03300</i> .	708
654	models: Found inside large language models. <i>arXiv</i>		
655	<i>preprint arXiv:2506.23235</i> .	Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel	709
		Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford,	710
656	Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harri-	Dario Amodei, and Paul F Christiano. 2020. Learn-	711
657	son Edwards, Bowen Baker, Teddy Lee, Jan Leike,	ing to summarize with human feedback. <i>Advances</i>	712
658	John Schulman, Ilya Sutskever, and Karl Cobbe.	<i>in neural information processing systems</i> , 33:3008–	713
659	2023. Let’s verify step by step. In <i>The Twelfth Inter-</i>	3021.	714
660	<i>national Conference on Learning Representations</i> .		
		Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang,	715
661	Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi,	Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett,	716
662	Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin.	and Andrea Zanette. 2024. Fast best-of-n decoding	717
663	2025. Understanding rl-zero-like training: A critical	via speculative rejection. <i>Advances in Neural Infor-</i>	718
664	perspective. <i>arXiv preprint arXiv:2503.20783</i> .	<i>mation Processing Systems</i> , 37:32630–32652.	719
665	Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler	Mirac Suzgun and 1 others. 2022. Challenging big-	720
666	Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon,	bench tasks and whether chain-of-thought can solve	721
667	Nouha Dziri, Shrimai Prabhunoye, Yiming Yang,	them. <i>arXiv preprint arXiv:2210.09261</i> .	722
668	and 1 others. 2023. Self-refine: Iterative refinement		
669	with self-feedback. <i>Advances in Neural Information</i>	Leitian Tao, Ilia Kulikov, Swarnadeep Saha, Tianlu	723
670	<i>Processing Systems</i> , 36:46534–46594.	Wang, Jing Xu, Sharon Li, Jason E Weston, and	724
		Ping Yu. 2025. Hybrid reinforcement: When re-	725
671	Samuel Marks and Max Tegmark. 2023. The geometry	ward is sparse, it’s better to be dense. <i>arXiv preprint</i>	726
672	of truth: Emergent linear structure in large language	<i>arXiv:2510.07242</i> .	727
673	model representations of true/false datasets. <i>arXiv</i>		
674	<i>preprint arXiv:2310.06824</i> .	Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le,	728
		Ed Chi, Sharan Narang, Aakanksha Chowdhery, and	729
675	Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida,	Denny Zhou. 2022. Self-consistency improves chain	730
676	Carroll Wainwright, Pamela Mishkin, Chong Zhang,	of thought reasoning in language models. <i>arXiv</i>	731
677	Sandhini Agarwal, Katarina Slama, Alex Ray, and 1	<i>preprint arXiv:2203.11171</i> .	732
678	others. 2022. Training language models to follow in-		
679	structions with human feedback. <i>Advances in neural</i>	Yiming Wang, Pei Zhang, Baosong Yang, Derek F	733
680	<i>information processing systems</i> , 35:27730–27744.	Wong, and Rui Wang. 2024. Latent space chain-of-	734
		embedding enables output-free llm self-evaluation.	735
		<i>arXiv preprint arXiv:2410.13640</i> .	736
681	Vardan Papyan, XY Han, and David L Donoho. 2020.	Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye,	737
682	Prevalence of neural collapse during the terminal	Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang,	738
683	phase of deep learning training. <i>Proceedings of</i>	Junjie Li, Ziming Miao, and 1 others. 2025. Rein-	739
684	<i>the National Academy of Sciences</i> , 117(40):24652–	forcement learning with verifiable rewards implicitly	740
685	24663.	incentivizes correct reasoning in base llms. <i>arXiv</i>	741
		<i>preprint arXiv:2506.14245</i> .	742
686	Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya		
687	Sutskever, and 1 others. 2018. Improving language	Zhangchen Xu, Yuetai Li, Fengqing Jiang, Bhaskar Ra-	743
688	understanding by generative pre-training.	masubramanian, Luyao Niu, Bill Yuchen Lin, and	744
		Radha Poovendran. 2025. Tinyv: Reducing false neg-	745
689	Rafael Rafailov, Archit Sharma, Eric Mitchell, Christo-	atives in verification improves rl for llm reasoning.	746
690	pher D Manning, Stefano Ermon, and Chelsea Finn.	<i>arXiv preprint arXiv:2505.14625</i> .	747
691	2023. Direct preference optimization: Your language		
692	model is secretly a reward model. <i>Advances in neural</i>	Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran,	748
693	<i>information processing systems</i> , 36:53728–53741.	Tom Griffiths, Yuan Cao, and Karthik Narasimhan.	749
		2023. Tree of thoughts: Deliberate problem solving	750
694	John Schulman, Filip Wolski, Prafulla Dhariwal,	with large language models. <i>Advances in neural</i>	751
695	Alec Radford, and Oleg Klimov. 2017. Proxi-	<i>information processing systems</i> , 36:11809–11822.	752
696	mal policy optimization algorithms. <i>arXiv preprint</i>		
697	<i>arXiv:1707.06347</i> .	Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan,	753
		Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan,	754
698	Adam Shai, Lucas Teixeira, Alexander Oldenziel, Sarah	Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo:	755
699	Marzen, and Paul Riechers. 2024. Transformers rep-		
700	resent belief state geometry in their residual stream.		

756 An open-source llm reinforcement learning system
757 at scale. *arXiv preprint arXiv:2503.14476*.

758 Yizhe Zhang, Jiatao Gu, Zhuofeng Wu, Shuangfei Zhai,
759 Joshua Susskind, and Navdeep Jaitly. 2023. Planner:
760 Generating diversified paragraph via latent language
761 diffusion model. *Advances in Neural Information
762 Processing Systems*, 36:80178–80190.

763 Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang,
764 Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen
765 Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023.
766 A survey of large language models. *arXiv preprint
767 arXiv:2303.18223*, 1(2).

768 Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen,
769 Xun Wu, Yaru Hao, Tengchao Lv, Shaohan
770 Huang, Lei Cui, Qixiang Ye, and 1 others. 2025.
771 Geometric-mean policy optimization. *arXiv preprint
772 arXiv:2507.20673*.

773 Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui
774 Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong
775 Liu, Rui Men, An Yang, and 1 others. 2025.
776 Group sequence policy optimization. *arXiv preprint
777 arXiv:2507.18071*.

778 Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan
779 Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin,
780 Zhuohan Li, Dacheng Li, Eric Xing, and 1 others.
781 2023. Judging llm-as-a-judge with mt-bench and
782 chatbot arena. *Advances in neural information pro-
783 cessing systems*, 36:46595–46623.

784 Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang,
785 Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin,
786 and Chao Du. 2025. Reinforcing general reasoning
787 without verifiers. *arXiv preprint arXiv:2505.21493*.

788 Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B
789 Brown, Alec Radford, Dario Amodei, Paul Chris-
790 tiano, and Geoffrey Irving. 2019. Fine-tuning lan-
791 guage models from human preferences. *arXiv
792 preprint arXiv:1909.08593*.

793 Andy Zou, Long Phan, Sarah Chen, James Campbell,
794 Phillip Guo, Richard Ren, Alexander Pan, Xuwang
795 Yin, Mantas Mazeika, Ann-Kathrin Dombrowski,
796 and 1 others. 2023. Representation engineering: A
797 top-down approach to ai transparency. *arXiv preprint
798 arXiv:2310.01405*.

799 A Usage of LLM

800 Large Language Models were used exclusively to
801 improve the clarity and fluency of English writing.
802 They were not involved in research ideation, ex-
803 perimental design, data analysis, or interpretation.
804 The authors take full responsibility for all content.

805 B Detailed Related Work

806 **Policy Optimization and Group-based Vari-
807 ants.**In early RLHF(Ziegler et al., 2019; Stien-
808 non et al., 2020; Ouyang et al., 2022; Lee et al.,

2023b) practices, PPO (Proximal Policy Optimiza-
809 tion)(Schulman et al., 2017) was the core algo-
810 rithm, stabilizing training through trust region con-
811 straints. However, PPO requires maintaining a
812 critic model of the same scale as the policy model,
813 introducing significant memory overhead. DPO
814 (Direct Preference Optimization)(Rafailov et al.,
815 2023) subsequently eliminated the need for crit-
816 ics by reparameterizing rewards as functions of
817 optimal policies, but its offline nature limits explo-
818 ration capabilities. To balance online exploration
819 and computational efficiency, GRPO (Group Rela-
820 tive Policy Optimization)(Shao et al., 2024) intro-
821 duced group-based advantage estimation and has
822 become a research focus. A series of variants have
823 emerged to address specific challenges: DAPO(Yu
824 et al., 2025) combines dynamic sampling with gra-
825 dient clipping for training stability, GSPO(Zheng
826 et al., 2025) employs sequence-level importance
827 sampling for MoE(Jacobs et al., 1991; Du et al.,
828 2022) models, Dr. GRPO(Liu et al., 2025) cor-
829 rects length and difficulty biases, and GMPO(Zhao
830 et al., 2025) uses geometric mean to resist reward
831 outliers. Despite significant progress at the opti-
832 mizer level, these methods have not addressed the
833 source of reward signals, still assuming the exis-
834 tence of perfect oracles (ground truth or expensive
835 external verifiers). Latent-GRPO is orthogonal to
836 these works: by extracting intrinsic geometric sig-
837 nals from the model, our reward mechanism can
838 be seamlessly integrated with any of these opti-
839 mization algorithms, achieving both “no external
840 supervision” and “efficient stable optimization.”
841 **Training-free evaluation Methods.**To eliminate
842 dependence on expensive labeled data, researchers
843 have explored training-free evaluation and self-
844 correction methods. Self-Consistency (SC)(Wang
845 et al., 2022) improves answer accuracy through ma-
846 jority voting over multiple reasoning paths. Self-
847 Refine(Madaan et al., 2023) enables iterative re-
848 finement by generating self-feedback without exter-
849 nal guidance. Tree-of-Thoughts (ToT)(Yao et al.,
850 2023) models problem-solving as tree search, ex-
851 ploring multiple reasoning branches and backtrack-
852 ing to find optimal solutions. Best-of-N(Stiennon
853 et al., 2020) sampling generates multiple candi-
854 dates and selects the best based on heuristics or con-
855 fidence scores. Forest-of-Thoughts(Bi et al., 2024)
856 extends ToT by integrating multiple trees with
857 sparse activation for efficiency. Speculative Re-
858 jection(Sun et al., 2024) accelerates Best-of-N by
859 early termination of low-quality candidates. While
860

these methods leverage various forms of “consensus” or “selection,” they primarily rely on discrete matching of final text results or explicit model outputs, failing to capture subtle semantic differences in reasoning processes. Moreover, they are typically used only during inference without converting to training signals. Unlike these approaches, Latent-GRPO does not rely on explicit text feedback but uses geometric structures in the latent space as implicit evaluation criteria. We demonstrate that geometric centroids in latent space contain richer semantic information than explicit text-based methods, providing denser and more robust gradient signals for RL training.

Latent Space and Latent Thinking. The latent space of LLMs is widely recognized to encode rich semantic and structural information (Zhang et al., 2023; Goyal et al., 2023; Hao et al., 2024; Geiping et al., 2025). Early probing work revealed that LLM hidden states contain linear directions representing syntax, sentiment, and truthfulness. Recent work has explored latent thinking, replacing verbose natural language steps with compact latent representations. CoE (Chain-of-Embedding) (Wang et al., 2024) uses progressive hidden states for output-free self-evaluation, showing significant differences between correct and incorrect answers without requiring training. LTO (Latent Thinking Optimization) (Du et al., 2025) and LaTRO (Chen et al., 2024) train additional latent classifiers (Latent Reward Models) to predict correctness and guide RL optimization, achieving 12.5% average accuracy improvement on GSM8K. However, these methods require training extra models, introducing computational overhead. EndoRM (Li et al., 2025) extracts “endogenous rewards” from LLM logits without training, but relies on output probabilities rather than latent geometric structures. Few works have attempted to directly convert geometric consensus in latent space into RL reward signals without additional training. Our work fills this gap by demonstrating that high-quality reward signals can be extracted from latent space through simple geometric methods. We establish that “Last Token Hidden States” combined with “Iterative Robust Centroid Estimation” effectively captures subtle semantic differences in reasoning, providing denser and more robust gradient signals than explicit text-based or probability-based methods.

C Detailed Analysis of Geometric-Based Scoring

We simulate the actual GRPO training scenario to validate whether geometric features can serve as reliable quality indicators. For each prompt, we generate a group of $G = 8$ trajectories (matching the typical GRPO group size) and compute geometric scores using our Iterative Robust Centroid Estimation (IRCE) algorithm. Figure 4 presents the effectiveness of geometric-based scoring:

Distribution Separability (Figure 4a). The box plot analysis reveals clear separation between different quality levels in the geometric reward space. Trajectories with high external scores (green, score > 0.9) exhibit highly concentrated distributions with median geometric scores close to 0 (near the centroid), while low-score trajectories (red, score < 0.3) show significantly right-skewed distributions with large variance (far from the centroid). The partial-correct trajectories (yellow, $0.3 < \text{score} < 0.9$) fall in between, demonstrating that geometric features provide fine-grained quality discrimination rather than binary classification.

Group-Level Ranking Consistency (Figure 4b-d). To validate the practical utility in GRPO’s advantage estimation, we examine three representative groups of 8 trajectories. The dual-axis plot shows ground truth scores (gray bars) and normalized geometric scores (blue line with markers). The Spearman rank correlation of 0.927 indicates high ranking consistency between geometric and external scores in the Figure 4b. Critically, the top-ranked samples identified by geometric scores (red circles) align well with ground truth, demonstrating that our method can effectively identify elite trajectories for policy optimization. In the Top-1 selection task across all groups, the consistency rate between geometric-based selection and external model selection exceeds 85%, confirming that geometric signals can effectively replace expensive external verifiers in GRPO’s advantage estimation.

D Detailed Introduction of Iterative Robust Centroid Estimation

For each prompt, the policy generates a group of G trajectories. From each trajectory, we extract the **terminal hidden state** $\mathbf{h}_i \in \mathbb{R}^d$ ($i = 1, \dots, G$), forming the sampled state matrix $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_G]^T \in \mathbb{R}^{G \times d}$. The Iterative Robust Centroid Estimation (IRCE) algorithm proceeds through the following steps:

Step 1: Spherical Projection. To eliminate magnitude fluctuations and ensure the distance metric focuses exclusively on semantic directionality, we apply L_2 normalization to project all terminal hidden states onto a unit hypersphere:

$$\tilde{\mathbf{h}}_i = \frac{\mathbf{h}_i}{\|\mathbf{h}_i\|_2} \quad (5)$$

Step 2: Initialize Centroid. The initial centroid $\boldsymbol{\mu}^{(0)}$ is computed as the normalized mean of the projected states:

$$\boldsymbol{\mu}^{(0)} = \frac{\frac{1}{G} \sum_{i=1}^G \tilde{\mathbf{h}}_i}{\left\| \frac{1}{G} \sum_{i=1}^G \tilde{\mathbf{h}}_i \right\|_2} \quad (6)$$

Step 3: Iterative Soft-Weighted Update. In each iteration $s = 0, 1, \dots, T-1$, we compute the Euclidean distance from each sample to the current centroid:

$$d_i^{(s)} = \|\tilde{\mathbf{h}}_i - \boldsymbol{\mu}^{(s)}\|_2 \quad (7)$$

To adapt to varying cluster densities across different training stages, we derive an adaptive scale parameter $\sigma^{(s)}$ based on the group distance distribution:

$$\sigma^{(s)} = \text{std}(\{d_1^{(s)}, \dots, d_G^{(s)}\}) + \epsilon \quad (8)$$

where ϵ is a small constant for numerical stability. We then calculate soft weights $w_i^{(s)}$ via a Gaussian kernel to diminish the influence of outliers:

$$w_i^{(s)} = \frac{\exp\left(-\frac{(d_i^{(s)})^2}{2(\sigma^{(s)})^2}\right)}{\sum_{j=1}^G \exp\left(-\frac{(d_j^{(s)})^2}{2(\sigma^{(s)})^2}\right)} \quad (9)$$

The centroid is updated using a weighted average and subsequently re-projected onto the hypersphere:

$$\boldsymbol{\mu}^{(s+1)} = \frac{\sum_{i=1}^G w_i^{(s)} \tilde{\mathbf{h}}_i}{\left\| \sum_{i=1}^G w_i^{(s)} \tilde{\mathbf{h}}_i \right\|_2} \quad (10)$$

Step 4: Reward Computation. Upon convergence after T iterations, the intrinsic reward R_i for each trajectory is defined as the negative Euclidean distance to the final centroid:

$$R_i = -\|\tilde{\mathbf{h}}_i - \boldsymbol{\mu}^{(T)}\|_2 \quad (11)$$

Optionally, we apply Min-Max normalization within the group to scale the rewards $R \in [0, 1]$, ensuring stable gradient estimation for the policy update.

Dataset	Total Size	Train	Val
GSM8K	8,500	2,000	1,000
MATH	12,500	3,000	1,000
Open-Platypus	20,726	4,500	2,000
<i>Open-Platypus Data Source Statistics:</i>			
MATH/PRM-800K	12,298	754	783
ReClor	4,530	693	437
Airoboros	2,605	528	251
ScienceQA	1,317	618	127
LeetCode Solutions	1,100	365	106
ARB	713	272	69
SciBench	616	249	59
TheoremQA	564	336	54
TigerBot Kaggle	386	193	37
OpenAssistant Guanaco	797	492	77
Total	20,726	4,500	2,000

Table 6: Overview of datasets used in RL training

E Usage of Datasets

E.1 Datasets Overview

We use three main datasets for our experiments, each serving different purposes in evaluating the generalization and robustness of Latent-GRPO. Table 6 provides a comprehensive overview of all three datasets, including their sources, sizes, and usage in our RL training.

E.2 Dataset Details

GSM8K is a dataset of 8,500 grade school math word problems designed to test multi-step mathematical reasoning. MATH is a challenging dataset containing 12,500 competition-level mathematics problems from high school and undergraduate mathematics competitions. Open-Platypus is a large-scale diverse instruction-following dataset focused on improving LLM logical reasoning skills, comprising 20,726 samples filtered from multiple sources (MATH/PRM-800K: 12,298, ReClor: 4,530, Airoboros: 2,605, ScienceQA: 1,317, LeetCode Solutions: 1,100, ARB: 713, SciBench: 616, TheoremQA: 564, TigerBot Kaggle: 386, OpenAssistant Guanaco: 797) using keyword search and Sentence Transformers (removing questions with similarity above 80%).

E.3 Benchmark

To comprehensively evaluate the performance of our model across general knowledge, logical reasoning, and advanced mathematical problem-solving, we select five representative benchmarks. These range from broad multi-task knowledge to extremely challenging competitive mathematics.

1029	MMLU Massive Multitask Language Understanding (Hendrycks et al., 2020) covers 57 subjects across STEM, the humanities, social sciences, and more. It tests both world knowledge and problem-solving capsules.	
1030		
1031		
1032		
1033		
1034	MATH-500 The MATH dataset (Hendrycks et al., 2021b) consists of 12,500 challenging high school math competition problems. Following recent literature (e.g., DeepSeek-R1), we use the MATH-500 subset, which comprises 500 representative problems from the test set to evaluate rigorous mathematical reasoning.	
1035		
1036		
1037		
1038		
1039		
1040		
1041	BBH Big-Bench Hard (BBH) (Suzgun et al., 2022) focuses on a subset of 23 challenging tasks from the BIG-bench suite where previous language models fell short of human multi-step reasoning capabilities.	
1042		
1043		
1044		
1045		
1046	AIME 24 & AIME 25 The American Invitational Mathematics Examination (AIME) is a prestigious high school competition. We evaluate our model on the AIME 2024 and the most recent AIME 2025 problems. These sets are particularly valuable as "out-of-distribution" (OOD) tests for reasoning models, as they require creative multi-step logical chains without standard templates.	
1047		
1048		
1049		
1050		
1051		
1052		
1053		
1054	F Experimental Hyperparameters	
1055	F.1 Hardware Configuration	
1056	All experiments are conducted on 1 GPU using PyTorch 2.6, CUDA 12.4 and the Transformers library 4.51.1. We employ bfloat16(Kalamkar et al., 2019) mixed precision training to improve memory efficiency and computational speed.	
1057		
1058		
1059		
1060		
1061	F.2 GRPO Training Configuration	
1062	All models use the Instruct version with Flash Attention 2 enabled. Maximum sequence length is 8192, generation temperature is 0.95, and Top-p sampling is 0.9. Group size is 8 trajectories per prompt, batch size is 1 prompt with mini-batch size 2. PPO inner update rounds are 1, clip ratio is 0.2, KL penalty coefficient is 0.1. Learning rate is 1×10^{-5} with linear warmup over 100 steps, maximum gradient norm is 1.0, weight decay is 0.01. We train for 2 epochs with logging every 10 steps and checkpoints every 100 steps. To simulate scenarios with constrained computational resources, the throughput for LLM-as-Judge scoring is restricted to 2 Queries Per Second (QPS).	
1063		
1064		
1065		
1066		
1067		
1068		
1069		
1070		
1071		
1072		
1073		
1074		
1075		
	F.3 IRCE Algorithm Configuration	1076
	The Iterative Robust Centroid Estimation algorithm uses 5 iterations with temperature parameter 0.5. Rewards are Min-Max normalized to $[0, 1]$ interval. Convergence threshold is 1×10^{-6} for early stopping. Epsilon for numerical stability is 1×10^{-8} .	1077
		1078
		1079
		1080
		1081
	G Detailed Reward Methods	1082
	G.1 Basemethods of reward	1083
	Rule-based reward For mathematical problems, we use symbolic computation (SymPy) to verify correctness by comparing the model’s final answer with the ground truth. For code generation tasks, we use sandboxed execution environments to test whether the generated code produces correct outputs. Rule-based reward provides binary rewards (0 or 1) based on exact correctness, making it a reliable but sparse reward signal. The main limitation is that it requires well-defined verification rules and cannot be applied to open-ended tasks like writing or summarization.	1084
		1085
		1086
		1087
		1088
		1089
		1090
		1091
		1092
		1093
		1094
		1095
	LLM-as-Judge. We use GPT-4o to evaluate response quality with a standardized prompt asking for binary correctness judgment. Each response is evaluated independently, and the judgment is converted to a binary reward (0 or 1). While LLM-as-Judge is more flexible and can handle diverse task types, it introduces external dependency, higher computational cost, and potential inconsistency in evaluation.	1096
		1097
		1098
		1099
		1100
		1101
		1102
		1103
		1104
	G.2 Latent Reward Methods	1105
	We compare four latent reward methods that extract rewards from hidden states without external supervision:	1106
		1107
		1108
	Mean Pool. Computes the centroid as the simple arithmetic mean of all normalized hidden states: $\mu = \frac{1}{G} \sum_{i=1}^G \tilde{\mathbf{h}}_i$. Rewards are computed as negative distances: $R_i = -\ \tilde{\mathbf{h}}_i - \mu\ _2$. This method is computationally efficient but sensitive to outliers.	1109
		1110
		1111
		1112
		1113
	K-Means. Applies K-Means clustering with $K = 2$ to group hidden states into quality and non-quality clusters. The algorithm iteratively optimizes cluster centers and assigns samples to nearest centers. Rewards are based on distance to the quality cluster center. This method identifies dominant quality patterns but requires parameter tuning and has higher computational overhead.	1114
		1115
		1116
		1117
		1118
		1119
		1120
		1121
	Eigen Centrality. Constructs a similarity matrix \mathbf{A} based on cosine similarity between hidden states. The principal eigenvector \mathbf{v} of \mathbf{A} is computed, and	1122
		1123
		1124

Benchmark	Domain	Size	Metric
MMLU	General Knowledge (57 subjects)	14,042	5-shot Accuracy
MATH-500	Competition Level Mathematics	500	Pass@1
BBH	Logical & Multi-step Reasoning	6,511	3-shot / CoT Accuracy
AIME 24	Advanced Mathematics Competition	30	Pass@k / Consensual
AIME 25	Advanced Mathematics Competition	30	Pass@k / Consensual

Table 7: Summary of evaluation benchmarks used in this work. For reasoning tasks, we prioritize pass@k and consensual evaluation to measure thinking capabilities.

component v_i serves as the reward for sample i . This method captures global graph structure but suffers from eigendecomposition computational bottleneck.

IRCE (Ours). Performs iterative soft-weighted centroid estimation. Step 1 normalizes hidden states to unit hypersphere. Step 2 initializes centroid as mean of normalized states. Step 3 iteratively updates centroid using soft weights based on Gaussian kernel. Step 4 computes final rewards as negative distances.

H Detailed Analysis of Experiments

H.1 Main Experiments

On GSM8K: Latent-GRPO achieves competitive or superior accuracy compared to LLM-as-Judge across all model scales. On Qwen-0.6B, Latent-GRPO reaches 61.25% versus 53.52% for LLM-as-Judge, while significantly reducing training time from 768.42m to 431.18m per epoch (**1.78× speed-up**). On Qwen-1.7B, Latent-GRPO outperforms LLM-as-Judge by a substantial margin (73.88% vs 64.20%) with **2.10× speed-up** (1032.55m vs 492.34m). On Qwen-4B, Latent-GRPO achieves 82.34% accuracy with **2.14× speed-up** (1411.72m vs 658.21m). Compared to Rule-based verification, Latent-GRPO consistently achieves higher accuracy while maintaining competitive training efficiency.

On MATH: Latent-GRPO consistently outperforms LLM-as-Judge across all model scales with substantial speed-ups. On Qwen-0.6B, Latent-GRPO improves from 52.94% to 58.47% while reducing time from 1224.15m to 718.63m (**1.70× speed-up**). On Qwen-1.7B, Latent-GRPO achieves 78.51% versus 65.77% with **1.98× speed-up** (1608.34m vs 811.51m). On Qwen-4B, Latent-GRPO achieves 77.53% versus 77.44% with **2.18× speed-up** (2357.31m vs 1081.47m), demonstrating that even when accuracy is comparable, significant computational-efficiency gains are achieved. Rule-based methods show weakness on Qwen-1.7B

(42.14%), where Latent-GRPO substantially outperforms.

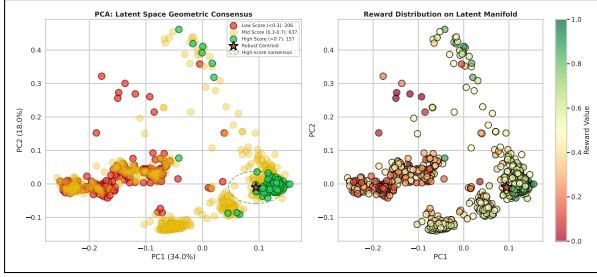
On Open-Platypus: Latent-GRPO demonstrates the strongest improvements on this diverse-reasoning benchmark. On Qwen-0.6B, Latent-GRPO achieves 40.56% versus 34.45% with **1.80× speed-up** (1937.82m vs 1079.27m). On Qwen-1.7B, Latent-GRPO reaches 64.82% versus 56.69% with **2.11× speed-up**. Most notably, on Qwen-4B, Latent-GRPO achieves 78.06% versus 65.21% with the highest speed-up of **2.16×** (3522.18m vs 1632.52m), demonstrating that intrinsic rewards are particularly effective for complex, diverse reasoning tasks where external verification becomes a critical computational bottleneck.

H.2 Ablation Experiments

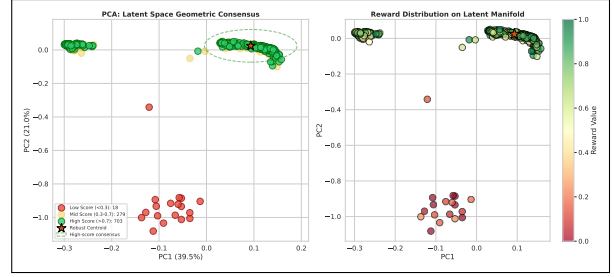
The ablation experiments systematically evaluate each component of Latent-GRPO through two critical design choices: hidden state extraction methods and latent score estimation algorithms.

Hidden State Extraction Methods As shown in Table 2, Last Token consistently achieves the highest accuracy across all three model scales: 61.25% on Qwen3-0.6B (2.51% improvement over Mean Pooling), 73.88% on Qwen3-1.7B (2.83% improvement), and 82.34% on Qwen3-4B (2.89% improvement). Notably, Weighted Mean (which selectively pools pre-designed key tokens) fails to substantially improve over Mean Pooling on any model scale (57.12% vs 58.74% on Qwen3-0.6B, 69.88% vs 71.05% on Qwen3-1.7B, 78.12% vs 79.45% on Qwen3-4B), suggesting that curating specific keyword positions cannot effectively identify which tokens carry reasoning quality signals.

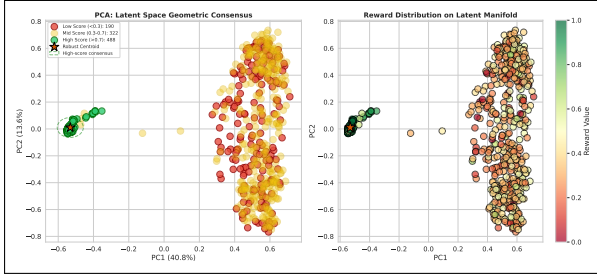
Latent Score Methods As shown in Table 3, IRCE achieves the highest accuracy on all three model scales: 61.25% on Qwen3-0.6B (outperforming Mean Pool by 4.13%), 73.88% on Qwen3-1.7B (outperforming Mean Pool by 5.43%), and 82.34% on Qwen3-4B (outperforming Mean Pool by 4.45%). IRCE also achieves the best computa-



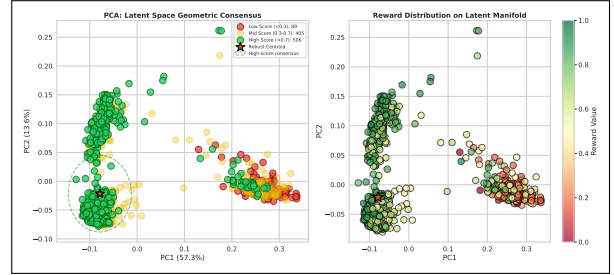
(a) Clustering of Qwen3-1.7B's 1000 rollouts on ScienceQA



(b) Clustering of Qwen3-4B's 1000 rollouts on ARBdata



(c) Clustering of the Qwen3-4B's 1000 rollouts on Mathdata



(d) Clustering of the Qwen3-4B's 1000 rollouts on ScienceQA

Figure 5: Visualization of latent manifold consensus across various model scales and benchmarks. Each panel shows the 2D PCA projection of 1,000 terminal hidden states. (a-d) consistently demonstrate that correct trajectories (green) cluster into a dense consensus core, while incorrect ones (red) are scattered.

1207 tional efficiency, with inference times of 431.18m
 1208 (0.6B), 492.34m (1.7B), and 658.21m (4B), sub-
 1209 stantially faster than K-Means (489.12m, 543.89m,
 1210 725.67m) and Eigen Centrality (468.76m, 531.42m,
 1211 708.45m).

1212 I Generalization of Latent-GRPO

1213 I.1 Generalization of Geometric Properties

1214 To evaluate the generalization of Geometric Prop-
 1215 erties, we choose additional data, including sci-
 1216 entific, logical reasoning, and more challenging
 1217 mathematical problems. Meanwhile we use mod-
 1218 els of more sizes to evaluate them. The geometric
 1219 clustering in latent space demonstrates consistent
 1220 patterns across different model sizes and datasets.
 1221 Each visualization is based on 1,000 rollouts from
 1222 a single test sample, with correctness scores as-
 1223 signed by GPT-4o. Correct trajectories (green cir-
 1224 cles, score > 0.7) form a visibly tight consensus
 1225 core around the truth centroid (gold star), while in-
 1226 correct trajectories (red circles, score < 0.3) scatter
 1227 as outliers. The dashed ellipse in each panel out-
 1228 lines the 95% confidence region of the consensus
 1229 core. In the original high-dimensional space, incor-
 1230 rect samples consistently lie farther from the cen-
 1231 troid than correct ones, quantifying the geometric
 1232 separability. This qualitative property generalizes

1233 across different model scales and benchmarks. As
 1234 shown in Figure 5, whether on Qwen3-1.7B with
 1235 ScienceQA or on Qwen3-4B with ARB and Math
 1236 data, the correct cluster remains compact while the
 1237 incorrect cloud disperses, indicating that geom-
 1238 etric proximity in latent space is a robust proxy for
 1239 reasoning quality. The separation pattern is stable
 1240 across configurations, suggesting that the underly-
 1241 ing geometric structure is universal rather than task-
 1242 or model-specific.

1243 J Case study

1244 Figure 6 provides the two cases between Latent-
 1245 GRPO and LLM-as-Judge method. These two
 1246 representative cases demonstrate that our Latent-
 1247 GRPO approach provides accurate continuous re-
 1248 ward signals during training. These signals are
 1249 endogenous in nature, requiring no external ver-
 1250 ifier supervision, and crucially, they are continuous
 1251 rather than binary like LLM-as-Judge. This contin-
 1252 uous signal not only accelerates iteration efficiency
 1253 through enhanced logical consistency, but also sta-
 1254 bilizes RL training by eliminating model collapse
 1255 caused by external verifier inconsistency and miti-
 1256 gating training loss from over-reliance on external
 1257 verifier accuracy.

Case Study 1	
<p>Query: John's cow weighs 400 pounds. It increased its weight to 1.5 times its starting weight. He is able to sell the cow for \$3 per pound. How much more is it worth after gaining the weight?</p>	
<p>Rollout #1</p> <p>The cow starts at 400 lbs. First, I need to calculate the new weight after it increases to 1.5 times..... $400 * 1.5 = 400 + 200 = 600$ lbs. Next, the value after weight gain is $600 \text{ lbs} * \\$3/\text{lb} = \\1800. The original value was $400 \text{ lbs} * \\$3 = \\1200. The question asks "how much more" it is worth, which means I need the difference: $\\$1800 - \\$1200 = \\$600$. Everything seems correct.</p> <p>The cow is worth \$600 more after gaining the weight.</p> <p>Latent Reward: 1.00 LLM-Judge: 1</p>	<p>Rollout #2</p> <p>I can solve this by finding the weight gain first. The weight increased from 1x to 1.5x, so the gain is 0.5x of the starting weight..... $400 \text{ lbs} * 0.5 = 200$ lbs gain. Since the price is \$3 per pound, the increased value is just the weight gain multiplied by the price: $200 \text{ lbs} * \\$3/\text{lb} = \\600. This is faster than calculating the total value.</p> <p>The cow is worth \$600 more.</p> <p>Latent Reward: 0.98 LLM-Judge: 1</p>
<p>Rollout #3</p> <p>Starting weight: 400 lbs. New weight: $400 * 1.5 = 600$ lbs..... Value after gain: $600 * 3 = 1800$. Original value: $400 * 3 = 1200$. Difference: $1800 - 1200 = 600$. The wording "how much more" is the key here. The cow is now worth 600 more than before.</p> <p>It's worth \$600 more.</p> <p>Latent Reward: 0.95 LLM-Judge: 1</p>	<p>Rollout #4</p> <p>$400 \text{ lbs} * 1.5 = 600$ lbs. Selling at \$3/lb makes the cow worth \$1800 now. It was worth $400 * 3 = \\$1200$ before the weight gain. The increase in value is $\\$1800 - \\$1200 = \\$600$. I'll confirm $400 * 1.5... 400 + 200 = 600$. Correct.</p> <p>The cow is worth \$600 more.</p> <p>Latent Reward: 0.92 LLM-Judge: 1</p>
<p>Rollout #5</p> <p>Wait, the cow gained 50% of its weight. 50% of 400 is 200. So the weight gain is 200 lbs. The price per pound is \$3, so the gain in value is $200 * 3 = \\$600$. This matches the method of subtracting total values.</p> <p>The value increased by \$600.</p> <p>Latent Reward: 0.88 LLM-Judge: 1</p>	<p>Rollout #6</p> <p>Let's see, the cow's initial weight is 400 pounds. It grows and becomes 1.5 times its weight, so $400 * 1.5 = 600$ pounds..... The sale price is \$3 per pound. So the total value of the cow after gaining the weight is $600 * 3... 600 \text{ times } 3 \text{ is } 1800$. So the cow is worth 1800 after the weight gain. That should be the answer.</p> <p>The cow's new value is \$1800.</p> <p>Latent Reward: 0.00 LLM-Judge: 0</p>
<p>Rollout #7</p> <p>Initial weight is 400. It increased its weight to 1.5 times, so I need $400 * 1.5... let's see, 1.5 \text{ times } 400... 400 \text{ plus } 1/4 \text{ of } 400 \text{ is } 500$ lbs..... New worth is $500 * \\$3 = \\1500. Original worth was $400 * \\$3 = \\1200. So the difference is $1500 - 1200 = 300$ dollars. Let me double check... yes, 300.</p> <p>The cow is worth \$300 more.</p> <p>Latent Reward: 0.15 LLM-Judge: 0</p>	<p>Rollout #8</p> <p>The cow weighs 400 lbs. 1.5x weight is 600 lbs. It sells for \$3/lb. Current price: $600 * 3 = 1800$. Previous price: $400 * 3 = 1200$. Difference: 600. Let me check if there's any other factor..... no, just "how much more"..... $1800 - 1200 = 600$. Correct.</p> <p>The cow is worth \$600 more.</p> <p>Latent Reward: 0.85 LLM-Judge: 1</p>
Case Study 2	
<p>Query: Let $\\$S$ be the sum of all positive integers $\\$n$ such that $\\$n^2+12n-2007$ is a perfect square. Find the remainder when $\\$S$ is divided by $\\$1000$.</p>	
<p>Rollout #1</p> <p>Let $\\$n^2+12n-2007=k^2$ for some integer $\\$k$. Completing the square on the left: $\\$(n+6)^2 - 36 - 2007 = k^2$, which simplifies to $\\$(n+6)^2 - k^2 = 2043$. This is a difference of squares: $\\$(n+6-k)(n+6+k) = 2043$. Factorizing 2043: $\\$2043 = 3 \times 681 = 3^2 \times 227$. Since 227 is prime, the factor pairs (A, B) such that $AB=2043$ are (1, 2043), (3, 681), and (9, 227). For each pair, $\\$n+6 = (A+B)/2$. This gives $\\$n = 1016, 336, 112$. All are positive. Sum $\\$S = 1016 + 336 + 112 = 1464$. $\\$1464 \pmod{\\$1000} = 464$.</p> <p>The remainder is 464.</p> <p>Latent Reward: 1.00 LLM-Judge: 1</p>	<p>Rollout #2</p> <p>We have $\\$n^2+12n-2007 = k^2$ implies $\\$(n+6)^2 - k^2 = 2043$. Let $\\$x = n+6$. Then $\\$(x-k)(x+k) = 2043$. We need to test factor pairs of $\\$2043 = 9 \times 227$. The pairs $\\$(d_1, d_2)$ must satisfy $\\$d_1 d_2 = 2043$ and $\\$d_2 > d_1$. Case 1: (1, 2043) $\\$to 2x = 2044, x = 1022$ $\\$to n=1016$. Case 2: (3, 681) $\\$to 2x = 684, x = 342$ $\\$to n=336$. Case 3: (9, 227) $\\$to 2x = 236, x = 118$ $\\$to n=112$. Sum $\\$S = 1016+336+112 = 1464$. The remainder is 464.</p> <p>The remainder is 464.</p> <p>Latent Reward: 0.98 LLM-Judge: 1</p>
<p>Rollout #3</p> <p>The equation is $\\$(n+6)^2 - k^2 = 2043$. The number of positive divisors of $\\$2043 = 3^2 \times 227$ is $\\$(2+1)(1+1) = 6$. This means there are $\\$6/2 = 3$ factor pairs. Each pair will give exactly one positive integer solution for $\\$n$. Calculating $\\$n$ for each: $\\$(2044/2)-6=1016$, $\\$(684/2)-6=336$, $\\$(236/2)-6=112$. Sum $\\$S = 1464$. The answer is 464.</p> <p>464.</p> <p>Latent Reward: 0.95 LLM-Judge: 1</p>	<p>Rollout #4</p> <p>Solving $\\$(n+6)^2 - k^2 = 2043$. Factor pairs of 2043: (1, 2043), (3, 681), (9, 227). Let $\\$(n+6) = d_1 + d_2$. This gives $\\$n+6 = 1022, 342, 118$. Thus $\\$n = 1016, 336, 112$. The sum of these values is 1464. Dividing 1464 by 1000 leaves a remainder of 464. Everything checks out.</p> <p>Remainder 464.</p> <p>Latent Reward: 0.91 LLM-Judge: 1</p>
<p>Rollout #5</p> <p>To find $\\$n$ such that $\\$n^2+12n-2007 = k^2$, we write $\\$(n+6)^2 - k^2 = 2043$. Factor pairs $\\$(x-k, x+k)$ of 2043: $\\$(1, 2043)$ $\\$to n=1016$; $\\$(3, 681)$ $\\$to n=336$; $\\$(9, 227)$ $\\$to n=112$. The sum is $\\$1016+336+112 = 1464$. Remainder mod 1000 is 464. No other solutions exist since $\\$n > 0$.</p> <p>464.</p> <p>Latent Reward: 0.86 LLM-Judge: 1</p>	<p>Rollout #6</p> <p>Setting the expression to $\\$k^2$ gives $\\$(n+6)^2 - k^2 = 2043$. Factorizing 2043, I find the divisors are 1, 3, 227, 681, 2043. The pairs are (1, 2043) and (3, 681). From (1, 2043), $\\$n+6 = 1022$ $\\$to n=1016$. From (3, 681), $\\$n+6 = 342$ $\\$to n=336$. I don't see any other factor pairs. So the sum $\\$S = 1016 + 336 = 1352$. The remainder is 352.</p> <p>The remainder is 352.</p> <p>Latent Reward: 0.20 LLM-Judge: 0</p>
<p>Rollout #7</p> <p>After factoring $\\$(n+6-k)(n+6+k) = 2043$, I solve for $\\$n$ using the pairs (1, 2043), (3, 681), and (9, 227). The values are $\\$n = 1016$, $\\$n = 336$, and $\\$n = 112$. Adding them together: $\\$1016 + 336 = 1352$. $\\$1352 + 112 = 1454$. Wait, let me re-add... $1352 + 112... \\$1300 + 100 = 1400$, $\\$52 + 12 = 64$... $\\$1464$. Oh, I'll stick with my first sum. The remainder is 1454.</p> <p>Remainder is 454.</p> <p>Latent Reward: 0.12 LLM-Judge: 1</p>	<p>Rollout #8</p> <p>I'll try small values for $\\$n$. If $\\$n=1$, $\\$1+12-2007 = -1994$ (not a square). If $\\$n=40$, $\\$1600+480-2007 = 73$. If $\\$n=50$, $\\$2500+600-2007 = 1093$. It doesn't look like there are many solutions. Maybe only very large $\\$n$ work, but I'll guess the sum is small or zero. Let's say 0.</p> <p>0.</p> <p>Latent Reward: 0.00 LLM-Judge: 0</p>

Figure 6: Case study of the reward between Latent-GRPO and LLM-as-Judge