DyJR: Preserving Local Policy Plasticity in Reinforcement Learning with Verifiable Rewards via Dynamic Jensen-Shannon Replay

Long Li[1,*] **Zhijian Zhou**[2,3,*] **Tianyi Wang**[4,*] **Weidi Xu**[5]
Zuming Huang[5] **Wei Chu**[5] **Zhe Wang**[1] **Shirui Pan**[1,†]
Chao Qu[2,6,†] **Yuan Qi**[2,6]

[1]Griffith University, Australia [2]Fudan University, China
[3]Shanghai Innovation Institute, China
[4]Tencent Youtu Lab, China
[5]INFLY TECH, China
[6]Shanghai Academy of Artificial Intelligence for Science, China
[*]Equal contribution. [†]Corresponding authors.
Correspondence: s.pan@griffith.edu.au, quchao@fudan.edu.cn

Abstract

While Reinforcement Learning (RL) enhances Large Language Model reasoning, on-policy algorithms like GRPO are sample-inefficient as they discard past rollouts. Existing experience replay methods address this by reusing accurate samples for direct policy updates, but this can incur additional storage costs and concentrate the policy on a narrow set of solutions. We argue that historical data should prioritize sustaining local policy plasticity rather than simply reinforcing accuracy. We define local policy plasticity as a persistent ability to remain backtrackable and switchable with respect to recent successful trajectories: throughout training, the policy maintains non-negligible support on recently reachable alternatives, so it can smoothly reallocate probability mass when needed without being locked into a single dominant path. To this end, we propose Dynamic Jensen-Shannon Replay (DyJR), a simple yet effective regularization framework using a dynamic reference distribution from recent trajectories. DyJR introduces two innovations: (1) A Time-Sensitive Dynamic Buffer that uses FIFO and adaptive sizing to retain only temporally proximal samples, synchronizing with model evolution; and (2) Jensen-Shannon Divergence Regularization, which replaces direct replay policy-gradient updates with a regularization signal that discourages premature over-commitment to a single Rank-1 trajectory. Experiments on mathematical reasoning, Text-to-SQL, and coding benchmarks demonstrate that DyJR improves over GRPO as well as baselines such as RLEP and Ex-GRPO, while maintaining training efficiency comparable to the original GRPO. Furthermore, Rank-k token probability evolution and semantic-mode analyses show reduced concentration and greater retention of alternative continuations. Matched admission–objective comparisons and repeated runs provide additional evidence for the roles of DyJR's components.

1 Introduction

Reasoning capability is central to the development of large language models (LLMs) [OpenAI, 2023, Yang et al., 2025, Zhu et al., 2025a]. Models such as OpenAI o1 demonstrate that reinforcement

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

learning (RL) can extend beyond preference alignment to improve reasoning through long chain-of-thought (CoT) exploration [OpenAI, 2024, Guo et al., 2025]. As "long thinking" with verifiable rewards becomes an increasingly important training paradigm [Chen et al., 2025b], a key bottleneck remains: on-policy reinforcement learning with verifiable rewards (RLVR) [Cai et al., 2025, Cai and Sugiyama, 2026] discards expensive rollout data after the current update [Shao et al., 2024, Zhan et al., 2025]. This limits reuse of past successes and increases the cost of scaling RL to long reasoning trajectories [Fu et al., 2025, Zhou et al., 2026]. Experience replay [Rolnick et al., 2019, Mnih et al., 2013, Lillicrap et al., 2015, Liu et al., 2026] offers a natural way to improve sample efficiency. Methods such as ReMix [Liang et al., 2025], RePO [Li et al., 2025b], Ex-GRPO [Zhan et al., 2025], and RLEP [Zhang et al., 2025] reuse historical trajectories for additional policy updates. In this data-augmentation view, replay contributes a second reward-optimization term alongside the online objective, schematically $\mathcal{J}(\theta) = \mathcal{J}_{\text{on}}(\theta) + \alpha\mathcal{J}_{\text{exp}}(\theta)$. Although admission rules and importance corrections differ across methods, the common purpose is to learn again from previously generated trajectories. We ask whether successful historical data can also serve a different role: maintaining access to useful alternatives as the current policy evolves.

Two challenges motivate this question. First, repeatedly reinforcing a restricted collection of successful trajectories can concentrate the policy on specific solution paths [Zhu et al., 2025b, Peng et al., 2025], reducing the alternatives available for further exploration [Wang et al., 2025, Yue et al., 2025, Cheng et al., 2025]. High reward alone does not reveal whether the policy still supports several ways of reaching a solution. Second, retaining a large history increases storage and mixes trajectories generated at different stages of training. Our training diagnostics reveal a *rapid transition window*: entropy decreases sharply during the early steps before the policy enters a more concentrated regime. A replay strategy must therefore provide enough successful trajectories during this transition while keeping its later reference relevant to the current policy. Based on these observations, we study experience replay as a regularization mechanism for sustaining *local policy plasticity*. By this term, we mean the persistent ability to remain backtrackable and switchable with respect to recent successful trajectories: the policy retains non-negligible support on nearby alternatives, so it can reallocate probability mass instead of becoming locked into a single dominant continuation. This does not require preserving every behavior encountered throughout training. Nor does a rise in token entropy alone establish that the retained alternatives correspond to useful reasoning strategies. We examine token dynamics together with separate semantic-mode measurements to assess this interpretation.

We propose *Dynamic Jensen-Shannon Replay* (DyJR), with two complementary innovations. Regarding **data construction**, a confidence-stratified admission rule selects successful trajectories for a first-in-first-out (FIFO) buffer. A higher admission rate populates the buffer during the initial transition, followed by a lower rate as training stabilizes. A short maximum-age window retains temporally adjacent samples and bounds the history used for regularization. Regarding **data utilization**, DyJR replaces additional replay reward-gradient updates with a Jensen-Shannon (JS) regularizer. Recent successful trajectories, together with their cached generation-time log-probabilities, provide a dynamic anchor that discourages those continuations from losing support. GRPO continues to learn from current rollouts, while replay helps preserve recently reachable alternatives. The distinction between construction and utilization is essential. Under a matched replay budget and freshness constraint, confidence-based admission improves the six-benchmark average by 2.5 points for off-policy replay and by 3.7 points for JS regularization. With this admission rule fixed, JS adds a further 1.5 points over off-policy replay. These controls support the complementary roles of admission and regularization rather than attributing the full gain to the divergence alone.

Our main contributions are threefold. **(1) Replay as plasticity regularization.** We introduce a replay framework that uses recent successful trajectories as a distributional reference, separating maintenance of local successful support from direct reinforcement of replayed rewards. **(2) Temporally local reference construction.** We combine confidence-stratified admission, a larger early admission rate, and a short FIFO window. The resulting reference adapts to policy evolution while avoiding storage of the full training history. **(3) Empirical evaluation and mechanism analysis.** We evaluate mathematics, Text-to-SQL, and coding with Qwen and Llama models, including repeated-run replay comparisons and matched mechanism controls. Rank-k token dynamics, sampling-budget scaling, and semantic-mode analyses provide complementary evidence about how data construction and utilization affect training.

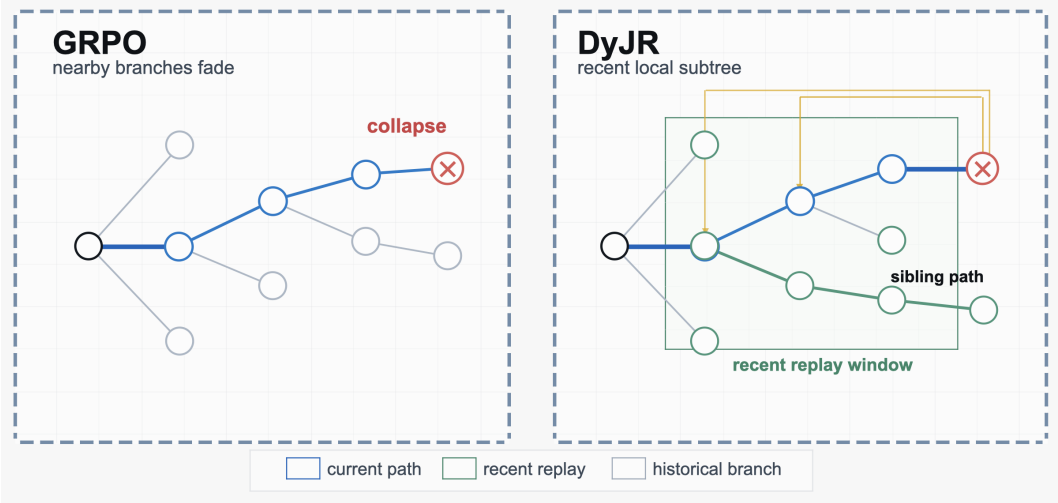


Figure 1: Conceptual illustration of local replay. DyJR retains recent successful branches as regularization targets when current sampling becomes concentrated. The tree illustrates accessible alternatives; the algorithm does not perform explicit tree search or a backtracking procedure.

2 Preliminaries

RLVR and GRPO. A policy π_θ generates a response y to prompt x and receives a verifiable reward $r(x, y)$. GRPO [Shao et al., 2024] samples G responses per prompt from $\pi_{\theta_{\text{old}}}$ and computes group-normalized advantages $\widehat{A}_i = (r(x, y_i) - \mu_x)/(\sigma_x + \varepsilon)$, where μ_x and σ_x are the group reward mean and standard deviation and ε is a numerical stabilizer. At token state $s_{i,j} = (x, y_i^{(<j)})$, let $\rho_{i,j} = \pi_\theta(y_i^{(j)} \mid s_{i,j})/\pi_{\theta_{\text{old}}}(y_i^{(j)} \mid s_{i,j})$. The clipped token loss is

$$\ell_{i,j}^{\text{GRPO}} = -\min\left[\rho_{i,j}\widehat{A}_i,\ \text{clip}(\rho_{i,j}, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}})\widehat{A}_i\right]. \tag{1}$$

We average over valid response tokens and omit the fixed-reference KL penalty.

Experience replay. Historical trajectories come from earlier policies. Direct replay uses them for reward-driven updates, whereas DyJR uses recent successes as regularization targets alongside the online GRPO loss. The online ratio uses this step's rollout policy; replay instead uses each trajectory's historical generating policy.

Jensen-Shannon divergence. For distributions P and Q, let $R = (P + Q)/2$. Jensen-Shannon divergence is $D_{\text{JS}}(P\|Q) = \frac{1}{2}D_{\text{KL}}(P\|R) + \frac{1}{2}D_{\text{KL}}(Q\|R)$. It is symmetric and bounded by $\ln 2$ with natural logarithms. DyJR uses a sampled surrogate with cached historical probabilities, detailed in Appendix B.2.

3 Method

DyJR combines a **Dynamic Replay Buffer** with **JS Regularization** to preserve recently successful alternatives as the policy evolves. Its two components address data construction and data utilization: the buffer selects a temporally local reference, and the regularizer uses that reference as a flexible anchor alongside the online GRPO objective. Figure 1 illustrates the mechanism; the full procedure appears in Algorithm 1.

Local Policy Plasticity. We use *local policy plasticity* to denote a *persistent* ability to remain *backtrackable and switchable* with respect to recent successful trajectories. The aim is to retain alternatives throughout training, so probability mass can shift when the current dominant continuation becomes over-committed. At a prefix state s, let a_k be the token with the k-th highest probability under the current policy, so a_1 is its dominant continuation. Let $q_t(\cdot \mid s)$ denote the conceptual

reference induced by recent successful trajectories. Our objective is to retain non-negligible support on replay-supported alternatives:

$$q_t(a_k \mid s) > 0,\ k \geq 2 \quad \Longrightarrow \quad \pi_{\theta_t}(a_k \mid s) \not\approx 0. \tag{2}$$

This notion concerns local adjustability: it neither requires coverage of every mode encountered during training nor merely asks the policy to change slowly. Equation 2 describes the intended behavior rather than a guarantee. Because token alternatives need not represent different reasoning modes, we assess both token-level diagnostics and semantic-mode analyses in Section 5.3.

3.1 Dynamic Reference Construction

A buffer entry stores a prompt, response, generation step τ, and the sampled response tokens' pre-update generating-policy log-probabilities ℓ_{old}. Before the update at step t, the usable buffer contains only previously admitted successful trajectories:

$$\mathcal{S}_t = \{(x, y, \ell_{\text{old}}, \tau) : \text{admitted at } \tau < t,\ r(x, y) = 1,\ t - \tau \leq M\}. \tag{3}$$

FIFO eviction enforces the maximum age M. The goal is to preserve *locally reachable alternatives* within the evolving policy neighborhood. Explicit semantic selection is difficult for long-chain reasoning, where surface-form similarity can misrepresent solution semantics; recency provides an inexpensive, task-agnostic proxy for policy relevance. Small clipped updates motivate this local view but do not guarantee bounded drift. A short window limits the exposure to stale anchors, with its practical effect assessed by the staleness ablation.

For each prompt x, define empirical confidence $C(x)$ as the number of correct responses among G current rollouts. We visit confidence levels from G down to 1, admitting only correct trajectories until a target fraction η_t of the N online rollouts is reached:

$$\mathcal{P}_t \leftarrow \mathcal{P}_t \cup \{(x, y, \ell_{\text{old}}, t) : C(x) = k,\ r(x, y) = 1\}, \quad k = G, G-1, \ldots, 1. \tag{4}$$

The target is a capacity budget, not a requirement to admit incorrect responses when successes are scarce. This high-to-low strategy first secures successful trajectories from high-confidence prompt groups, then relaxes the confidence criterion when these do not fill the budget. Compared with admitting only fully successful groups, it can therefore include rare successful trajectories from harder prompts. This deliberately selected reference remains biased toward successful responses; it is not an unbiased estimate of the rollout distribution. We use $\eta_t = 20\%$ for the first 20 training steps and 5% thereafter, with $M = 8$. Early RLVR training can sharply concentrate the policy distribution. The larger initial rate helps populate the buffer during this rapid transition, retaining successful early alternatives before they become difficult to sample. The admission rates and M were selected in initial pilots and reused for the 1.7B model without retuning. Section 4.3 distinguishes the evidence for early buffer population from the storage benefit of later contraction.

3.2 JS Regularization Implementation

We implement a sampled, token-level JS-style objective [Wang et al., 2023]. For replay trajectory s and response position j, define

$$u_s^{(j)} = \exp\left[\log \pi_\theta(y_s^{(j)} \mid x_s, y_s^{(<j)}) - \ell_{\text{old},s}^{(j)}\right], \tag{5}$$

$$f(u) = u \log u - (u+1) \log \frac{u+1}{2}, \tag{6}$$

$$\mathcal{L}_{\text{JS}}(\theta) = \frac{\sum_{s \in B_{\text{replay}}} \sum_j m_s^{(j)} f(u_s^{(j)})}{\sum_{s \in B_{\text{replay}}} \sum_j m_s^{(j)}}. \tag{7}$$

Here B_{replay} denotes replay rows in the current optimization micro-batch, and $m_s^{(j)}$ is its valid-response-token mask. Aggregation is token-mean within this masked batch, with zero replay contribution if it contains no valid replay tokens; cached log-probabilities remain fixed. This avoids re-evaluating the historical generating policy. The regularizer uses the generating policy associated with each trajectory; it does not explicitly construct or evaluate the full mixture density q_t.

The displayed generator gives $\mathbb{E}_{z \sim Q} f(P(z)/Q(z)) = 2D_{\text{JS}}(P\|Q)$ under exact reference sampling, with its normalization absorbed into the coefficient. Success filtering and cached per-policy denominators make this a surrogate rather than an unbiased mixture-divergence estimate. Bounded JS does not bound individual sample terms or gradients; Appendix B.2 details these distinctions.

3.3 Joint Optimization Objective

GRPO encourages higher-reward solutions while replay regularization anchors the policy to recent successful alternatives. The joint objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{GRPO}} + \alpha_{\text{JS}} \mathcal{L}_{\text{JS}}. \tag{8}$$

We use $\alpha_{\text{JS}} = 0.05$. Our tuning rule keeps the online policy loss and weighted JS loss at roughly a 10:1 scale. Starting with a short recency window, we adjust α_{JS} so the anchor remains about one order of magnitude weaker than the online objective. This balances an ineffective anchor against over-constraining policy updates; it is a loss-scale heuristic, not a fixed relationship between gradients.

Each step generates 4,096 online trajectories from 512 prompts with eight responses each. If the buffer contains at least 512 historical trajectories, we sample 512 uniformly without replacement once and hold them fixed throughout the step; otherwise replay is skipped. One epoch over the 4,608 shuffled rows, with mini-batch size 128, gives 36 optimizer steps. Online and replay rows contribute to GRPO and JS, respectively. Current successful trajectories enter the buffer only after optimization, retaining their rollout-time log-probabilities. Algorithm 1 gives the complete ordering and batching details.

4 Experiments

4.1 Tasks, Baselines, and Evaluation

We evaluate three task families. **Mathematics:** Qwen3-4B-Base is trained on Reinforce-Ada-Hard, a 14k-problem subset of OpenR1-Math-220k [Xiong et al., 2025], with responses up to 6,144 tokens. We evaluate AIME25, AMC23, Beyond AIME, BRUMO25, HMMT25, and Minerva, and also test Qwen3-1.7B-Base under sparser rewards. **Text-to-SQL:** Llama-3.1-8B-Instruct is trained on BIRD and evaluated on BIRD and Spider [Li et al., 2024b, Yu et al., 2018]. **Coding:** Qwen2.5-7B-Instruct is trained on CodeContests and evaluated on LiveCodeBench-V2 and held-out CodeContests. Appendix D gives configurations and metrics.

Baselines are GRPO without the reference-KL penalty, DAPO [Yu et al., 2025], DPH-RL [Li et al., 2025a], RLEP [Zhang et al., 2025], and Ex-GRPO [Zhan et al., 2025]. All are reproduced in the same VERL training stack, retaining their method-specific replay settings. We use an upper clipping parameter of 0.28 throughout. The main mathematical runs use AdamW with a learning rate of 2×10^{-5}, weight decay 0.1, and gradient clipping at 1.0. Each run uses 32 A800 GPUs for approximately 20 hours. Unless stated otherwise, ablations change the specified component of DyJR under the same training budget. Checkpoints are evaluated every 80 steps on the six mathematical benchmarks, without a separate validation split. Selecting the highest six-benchmark average yields the same final checkpoint, step 240, for every method and run in the main comparison. Thus, all reported results use a common endpoint. GRPO, DyJR, RLEP, and Ex-GRPO have three independent runs in the 4B comparison; other rows and ablations are single runs unless specified. Appendix H.1 reports the run-level results and sample standard deviations across three runs.

4.2 Mathematical Reasoning

Table 1 shows that DyJR improves the six-benchmark average from GRPO's 29.8 to 34.1, a gain of 4.3 percentage points. It also exceeds Ex-GRPO (32.6) and RLEP (31.8). Appendix H.1 gives benchmark-level variability and all newly repeated baseline runs. The remaining single-run ablations do not establish statistical significance. In the weaker Qwen3-1.7B setting, DyJR improves the average from 17.3 to 19.5. We reuse $M = 8$, the 20%/5% admission schedule, the replay-to-online ratio, and the loss-scale heuristic rather than retuning these defaults. This demonstrates transfer between the tested sizes without establishing universally optimal defaults.

4.3 Separating Admission from Replay Utilization

The matched comparison in Table 2 isolates two complementary choices. Confidence-based admission improves PG+IS by 2.5 points and JS by 3.7 points. With this admission rule fixed, JS improves over PG+IS by 1.5 points; under perfect-only admission, the difference is only 0.3 points. Adequate

Table 1: Mathematical reasoning accuracy (%). Minerva uses Mean@16; the other benchmarks use Mean@256. The 4B GRPO, DyJR, RLEP, and Ex-GRPO rows show three-run means; other rows are single runs. Run-level variability is reported in Appendix H.1. Bold marks the best main-method result within each model size.

Method	AIME25	AMC23	Beyond AIME	BRUMO25	HMMT25	Minerva	Avg
Qwen3-4B-Base							
Base model	8.2	51.5	4.3	16.6	1.9	29.0	18.6
GRPO	19.1	64.6	10.7	29.6	9.8	45.2	29.8
DAPO	21.5	65.5	10.6	30.4	9.2	47.8	30.8
DPH-RL	22.8	65.2	10.8	30.1	10.5	48.1	31.3
RLEP	22.6	70.9	10.9	28.7	10.7	47.3	31.8
Ex-GRPO	22.2	69.9	11.4	33.0	10.0	49.2	32.6
DyJR	**23.1**	**72.0**	**12.7**	**33.4**	**12.7**	**50.4**	**34.1**
Qwen3-1.7B-Base (sparse rewards)							
GRPO	8.5	47.8	3.1	11.7	**1.6**	31.3	17.3
DyJR	**10.1**	**53.8**	**3.9**	**14.4**	**1.6**	**33.4**	**19.5**

Table 2: Matched admission–objective comparison at $M = 8$. Values are six-benchmark averages (%). The training budget, replay batch size, schedule, and freshness constraint are fixed. PG+IS denotes off-policy policy gradients with importance ratios.

Admission rule	PG+IS	JS regularization
Perfect-only	30.1	30.4
Confidence-based	32.6	**34.1**

buffer population is therefore an important prerequisite in the sparse-success setting. The auxiliary controls are single runs; their comparison with the reported DyJR mean supports a conditional benefit of JS, rather than a decomposition that assigns the entire DyJR–GRPO gap to the regularizer.

The admission schedule also separates early availability from later storage cost. Constant 20% admission yields 34.2, close to 34.1 for the 20%→5% schedule. Reducing initial admission to 10% or 5% lowers the average to 32.4 or 31.3. Thus, later contraction reduces admission and capacity without an observed performance loss in this comparison, while insufficient early population is harmful. It does not establish that late trajectories are intrinsically less valuable (Appendix E).

4.4 Reference, Divergence, and Entropy Controls

Table 3 compares reference construction and divergence choices. Holding the replay budget, admission rule, and coefficient fixed, Static Ref + JS obtains 31.7 versus 34.1 for DyJR. DPH-RL, another JS-based method, obtains 31.3. These comparisons emphasize reference construction rather than identifying the divergence alone as the source of improvement. Reverse-KL to recent historical anchors obtains 29.4, so conservative anchoring alone is insufficient in this setting. Replacing JS with forward KL yields 32.5. Together these results favor the tested combination of a recent reference and JS-style utilization, without constituting a general theorem about either divergence.

Standard token-level entropy regularization yields 29.7, compared with 29.8 for GRPO and 34.1 for DyJR. The separately evaluated dynamic entropy bonus obtains 30.3; these are distinct controls (Appendix H.3). An undirected entropy term need not preferentially preserve successful alternative continuations. For recency, increasing M from 8 to 16, 32, or 64 reduces the average to 32.9, 31.7, or 30.9. The effect of a longer window also depends on admission: with perfect-only selection, increasing M from 8 to 32 raises the average from 30.4 to 31.3, whereas confidence-based selection falls from 34.1 to 31.7 under the same change (Appendix F). A longer history can alleviate scarcity under a restrictive admission rule, while older anchors may be less useful once enough recent successes are available. This interaction motivates choosing admission and freshness together; neither

Table 3: Ablations of the replay reference, divergence, regularization coefficient, and recency window. Values are mathematical accuracy (%); other settings follow DyJR. The default is $\alpha_{\mathrm{JS}} = 0.05$ and $M = 8$. Controls are single runs; the DyJR row shows its three-run mean.

Setting	AIME25	AMC23	Beyond AIME	BRUMO25	HMMT25	Minerva	Avg
DyJR (default)	23.1	72.0	12.7	33.4	12.7	50.4	**34.1**
Divergence and reference construction							
Forward KL	22.8	70.7	11.3	32.6	10.5	47.3	32.5
Static Ref + JS	22.7	66.3	11.2	29.9	10.7	49.4	31.7
Reverse-KL (recent)	18.7	64.3	10.1	28.8	10.1	44.3	29.4
Regularization coefficient and recency window							
$\alpha_{\mathrm{JS}} = 0.01$	22.5	69.3	11.5	31.0	12.5	49.5	32.7
$\alpha_{\mathrm{JS}} = 0.2$	21.4	66.7	10.1	29.0	10.9	47.4	30.9
$M = 16$	23.6	69.3	11.3	31.0	13.6	48.8	32.9
$M = 32$	22.0	68.9	10.9	30.2	10.3	47.7	31.7
$M = 64$	21.3	67.9	10.5	29.3	9.7	46.9	30.9

Table 4: Cross-domain results (%). SQL columns report execution-based Pass@1 and Pass@16. Coding columns report Pass@16 under unit-test verification. These are single-run comparisons. Bold marks the best value in each column.

	BIRD		Spider		Coding (Pass@16)	
Method	Pass@1	Pass@16	Pass@1	Pass@16	LiveCodeBench V2	CodeContests
GRPO	59.4	68.4	72.5	80.3	30.9	25.5
DAPO	60.1	69.2	71.5	79.4	30.7	26.4
DPH-RL	**62.8**	72.4	76.0	84.1	31.2	26.7
DyJR	62.7	**72.9**	**77.5**	**87.3**	**31.8**	**27.2**

a short window nor a large buffer is uniformly preferable across these controls. For regularization strength, coefficients 0.01 and 0.2 yield 32.7 and 30.9. The latter is an intentionally strong stress-test setting, still above GRPO, rather than evidence of a sharp instability boundary.

4.5 Transfer to SQL and Coding

DyJR improves over GRPO on both SQL benchmarks and both coding benchmarks (Table 4). On BIRD and Spider, Pass@1 increases by 3.3 and 5.0 points, and Pass@16 by 4.5 and 7.0 points. DPH-RL is slightly higher on BIRD Pass@1 (62.8 versus 62.7), while DyJR has the highest values in the other reported columns. Coding gains over GRPO are 0.9 and 1.7 points. These results extend the evaluation beyond competition mathematics; their single-run nature limits conclusions about smaller gaps.

5 Analysis

5.1 Token-Level Training Dynamics

Following prior token-probability diagnostics [Peng et al., 2025], we track a top-k-normalized entropy and the average probability of each ranked token. For top-k log-probabilities ℓ_i, define $\widetilde{p}_i = \exp(\ell_i)/\sum_{j=1}^{k}\exp(\ell_j)$ and $H_{\text{top-}k} = -\sum_i \widetilde{p}_i \log \widetilde{p}_i$. This is entropy within the retained token set, not full-vocabulary entropy. The average rank-k probability is $\overline{P}_{\text{rank-}k} = N^{-1}\sum_{n=1}^{N}\exp(\ell^{(n)}_{\text{rank-}k})$.

In Figure 2, GRPO rapidly concentrates probability on rank-1 tokens and remains highly concentrated. DyJR also adapts sharply at the start but later assigns more mass to rank-2 and rank-3

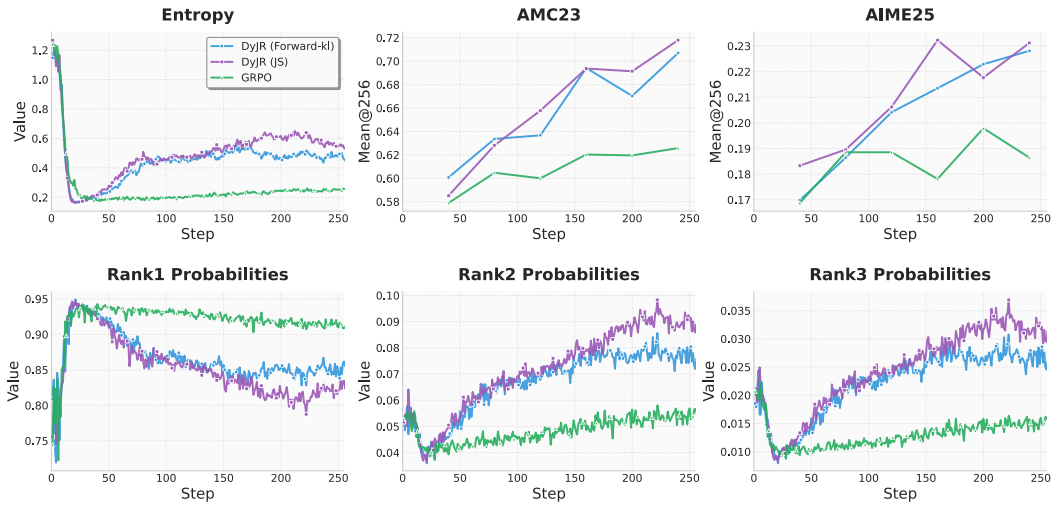


Figure 2: Qwen3-4B-Base training dynamics: top-k-normalized entropy, average ranked-token probabilities, and evaluation accuracy. Token concentration is an indirect diagnostic of the local-plasticity interpretation, rather than a measurement of semantic modes.

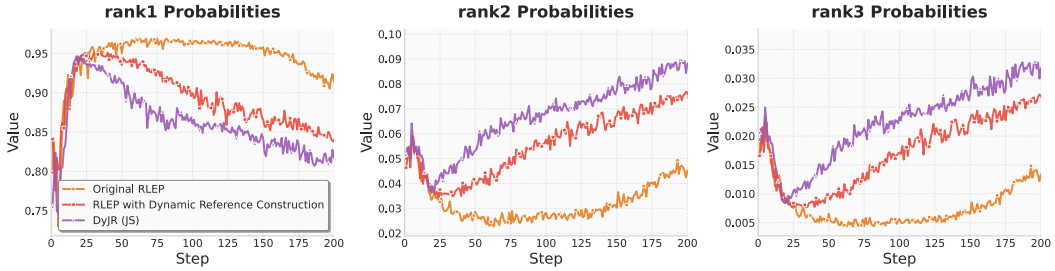


Figure 3: Rank-k probability trends for RLEP, RLEP with dynamic reference construction, and complete DyJR.

alternatives. The forward-KL variant lies between the two in several diagnostics and remains below JS in the reported accuracy comparisons. These patterns are consistent with reduced token-level concentration under DyJR. They do not establish that each additional token alternative is useful, or that token concentration alone causes a performance plateau.

5.2 From RLEP to DyJR

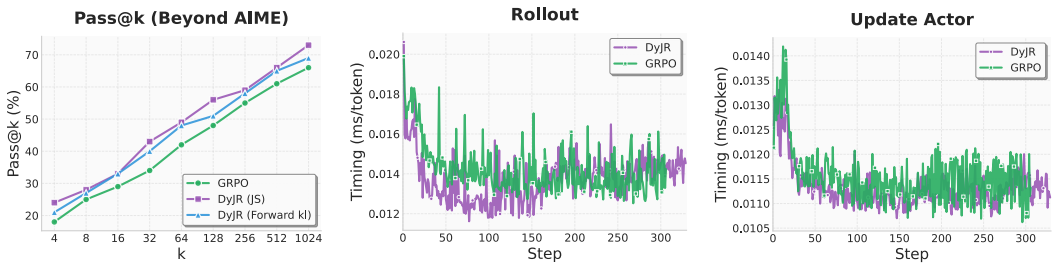
Figure 3 compares original RLEP, a variant that replaces its sampling mechanism with DyJR's dynamic reference construction while retaining replay policy updates, and complete DyJR. Dynamic construction reduces rank-1 concentration, and JS utilization further shifts mass toward alternatives. The construction variant changes more than a single scalar hyperparameter; the matched admission–objective experiment in Table 2 provides a more direct comparison of those two choices. We interpret the figure as a descriptive account of their training dynamics rather than a causal isolation of JS.

5.3 Semantic Modes Beyond Token Statistics

We supplement these proxies with two protocol-specific analyses. In controlled NL2SQL, supervised fine-tuning constructs five solution families with identifying prefixes. With 32 responses per question, 95% of questions yield only one family after GRPO, whereas 89% yield at least two after DyJR. Prefixes label predefined protocols; they do not independently verify the meaning or correctness of an unconstrained response.

Table 5: Mathematical strategy diversity on the same 50 problems, with 16 complete derivations per problem and model. Columns 2–6 give the percentage of problems. Modes include coherent incorrect strategies; the final column is the reported mean number of modes per problem.

Model	1 mode	2 modes	3 modes	≥ 4 modes	≥ 2 modes	Mean modes
Qwen3-4B-Base	22%	24%	34%	20%	78%	2.6
GRPO	88%	4%	6%	2%	12%	1.3
DyJR	42%	18%	24%	16%	58%	2.2



(a) (b) (c)

Figure 4: (a) Pass@k performance on the **Beyond AIME** dataset across varying sampling budgets k. (b) and (c) Computational latency comparison between GRPO (w/o KL) and DyJR during the *Rollout* phase and the *Actor Update* phase (encompassing log-probability calculation, forward passes, and gradient updates), respectively.

Table 5 reports the mathematical strategy distribution. We fix 50 problems and sample 16 complete derivations per problem and model. GPT-5.6-sol clusters executed mathematical strategies using all responses and their correctness labels, merging surface variation and retaining coherent incorrect attempts. Differences in wording, calculation order, arithmetic errors, and minor reparameterizations are merged when the core strategy is unchanged; responses without an identifiable strategy are excluded. Thus, the count measures distinct mathematical approaches rather than surface forms, and does not require every identified approach to contain a correct solution. The proportion of problems with at least two modes is 78% for Qwen3-4B-Base, 12% for GRPO, and 58% for DyJR; mean mode counts are 2.6, 1.3, and 2.2. DyJR therefore retains more observed strategy diversity than GRPO on this set, while remaining below the base model. The small sample, judge dependence, and inclusion of incorrect strategies limit generalization. Appendix H.4–H.5 gives the complete protocols and SQL distribution.

5.4 Sampling Budget and Computational Cost

On Beyond AIME, DyJR's Pass@k continues improving as the sampling budget increases to 1,024 and exceeds GRPO and the forward-KL variant in the reported curves (Figure 4). This provides a task-level complement to the token diagnostics. Unlike token entropy, Pass@k measures whether a sampling budget uncovers at least one verified solution. Its improvement is therefore relevant to the usefulness of retained alternatives, although the curve alone cannot distinguish better individual samples from less correlated attempts or identify a causal mechanism. The same figure reports rollout and actor-update latency, which remain close to the GRPO baseline in the measured configuration; Appendix I gives further details.

Replay storage depends on the recent admission window: $|\mathcal{S}_t| \leq \sum_{\tau=t-M}^{t-1} \eta_\tau N$, up to integer rounding and the supply of successful trajectories. At $N = 4,096$ and $M = 8$, the 20% warm-up rate allows about 6,554 trajectories; after the window contains only 5% admission steps, capacity is about 1,638. Thus, the steady-state bound is smaller than the warm-up bound and independent of the total training-set size. Token IDs require integer storage; cached log-probabilities and masks add further memory. Table 6 reports an isolated four-GPU token-ID benchmark: DyJR allocation stays constant across query counts, while RLEP's filled per-query capacity scales with them. Appendix H.6 gives the complete measurement protocol. It is not a measurement of complete training

Table 6: Measured resident-memory increase (GiB), summed over four H100 GPUs, in an isolated token-ID storage benchmark. Each trajectory has 8,192 `int32` token IDs. RLEP is tested at maximum per-query capacity; DyJR uses a fixed total capacity. Values include allocation/runtime overhead and exclude other replay fields and training state.

Storage configuration	$Q = 10$K	$Q = 30$K	$Q = 50$K	$Q = 100$K
RLEP: 64 trajectories/query	19.91	59.31	99.03	195.86
DyJR: 1,638 trajectories total	0.41	0.41	0.41	0.41

memory. The 512 replay rows add forward/backward work at a 1:8 replay-to-online ratio; their practical cost depends on implementation and sequence lengths.

6 Related Work

Experience Replay in LLM Training. RLVR supplies objective feedback in mathematics and coding [Lambert et al., 2025, Shao et al., 2024, Team, 2024, Li et al., 2024c]. Building on reasoning systems such as OpenAI o1 and DeepSeek-R1 [OpenAI, 2024, Guo et al., 2025], QwQ [Team, 2025] and Kimi k1.5 [Team et al., 2025] further explore this training paradigm. These developments make rollout reuse particularly relevant for long reasoning trajectories: collecting each response is costly, while the training distribution shifts as the policy improves. Originally developed for off-policy reinforcement learning [Mnih et al., 2013, Lillicrap et al., 2015], experience replay has also been studied in continual learning [Rolnick et al., 2019] and adapted to LLM training to improve sample efficiency. ReMix [Liang et al., 2025] and RePO [Li et al., 2025b] combine on-policy and off-policy learning, while RLEP [Zhang et al., 2025] and Ex-GRPO [Zhan et al., 2025] focus on reusing high-quality historical trajectories. ARPO [Lu et al., 2025] likewise integrates replay with GRPO. These approaches motivate careful treatment of the mismatch between current and historical policies. DyJR uses recent replay as a regularization reference rather than an additional reward-gradient dataset. Our matched controls distinguish the benefit of constructing a populated, recent reference from the incremental benefit of JS-based utilization.

Policy Plasticity Maintenance in LLM RL. Rapid concentration during RLVR has motivated work on entropy, exploration, and distributional constraints [Agarwal et al., 2025, Cheng et al., 2025, Tan et al., 2025]. Entropy-based techniques [Wang et al., 2025, Chen et al., 2025a] modify the training signal to retain exploratory behavior. SimKO [Peng et al., 2025] and W-REINFORCE [Zhu et al., 2025b] differentiate the treatment of successful and unsuccessful samples to counter distribution narrowing. DPH-RL [Li et al., 2025a] studies the f-divergence family as an alternative to reverse KL. DyJR complements these directions through a selectively populated, temporally refreshed reference: its regularizer targets recently successful continuations instead of increasing uncertainty indiscriminately. A fixed initial policy supplies the same reference throughout training, whereas temporally local replay tracks successful behaviors reached by intervening policies. The reverse-KL and static-reference controls in Section 4.4 compare these alternatives; their results caution against treating all regularization toward past policies as equivalent. Additional RLVR background appears in Appendix C.

7 Conclusion

We introduced DyJR, a framework that uses experience replay to sustain local policy plasticity during RLVR. A time-sensitive buffer retains recent successful trajectories, and JS regularization uses them as a dynamic anchor alongside the on-policy objective. Experiments on mathematics, Text-to-SQL, and coding demonstrate improvements over the tested baselines. Matched controls show that both adequate buffer admission and the choice of replay objective matter, while a larger early admission rate followed by contraction limits storage without an observed loss in the reported comparison. Rank-token dynamics and semantic-mode analyses support the interpretation that DyJR retains more alternatives around the current policy. These measurements do not establish a universal mechanism, and extending the approach to larger models and more direct behavioral criteria for replay remains an important direction.

Funding and Competing Interests

This research received no external funding. The authors declare no competing interests.

References

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning, 2025. URL https://arxiv.org/abs/2505.15134.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL https://matharena.ai/.

Xin-Qiang Cai and Masashi Sugiyama. Vi-curl: Stabilizing verifier-independent rl reasoning via confidence-guided variance reduction. *arXiv preprint arXiv:2602.12579*, 2026.

Xin-Qiang Cai, Wei Wang, Feng Liu, Tongliang Liu, Gang Niu, and Masashi Sugiyama. Reinforcement learning with verifiable yet noisy rewards under imperfect verifiers. *arXiv preprint arXiv:2510.00915*, 2025.

Minghan Chen, Guikun Chen, Wenguan Wang, and Yi Yang. Seed-grpo: Semantic entropy enhanced grpo for uncertainty-aware policy optimization, 2025a. URL https://arxiv.org/abs/2505.12346.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*, 2025b.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. Reasoning with exploration: An entropy perspective on reinforcement learning for llms, 2025. URL https://arxiv.org/abs/2506.14758.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, Tongkai Yang, Binhang Yuan, and Yi Wu. Areal: A large-scale asynchronous reinforcement learning system for language reasoning, 2025. URL https://arxiv.org/abs/2505.24298.

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

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

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

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024a.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36, 2024b.

Long Li, Xuzheng He, Haozhe Wang, Linlin Wang, and Liang He. How do humans write code? large models do it the same way too. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4638–4649, Miami, Florida, USA, November 2024c. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.267. URL https://aclanthology.org/2024.emnlp-main.267/.

Long Li, Jiaran Hao, Jason Klein Liu, Zhijian Zhou, Yanting Miao, Wei Pang, Xiaoyu Tan, Wei Chu, Zhe Wang, Shirui Pan, et al. The choice of divergence: A neglected key to mitigating diversity collapse in reinforcement learning with verifiable reward. *arXiv preprint arXiv:2509.07430*, 2025a.

Siheng Li, Zhanhui Zhou, Wai Lam, Chao Yang, and Chaochao Lu. Repo: Replay-enhanced policy optimization. *arXiv preprint arXiv:2506.09340*, 2025b.

Jing Liang, Hongyao Tang, Yi Ma, Jinyi Liu, Yan Zheng, Shuyue Hu, Lei Bai, and Jianye Hao. Squeeze the soaked sponge: Efficient off-policy reinforcement finetuning for large language model. *arXiv preprint arXiv:2507.06892*, 2025.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015. URL https://arxiv.org/abs/1509.02971.

Weiting Liu, Han Wu, Yufei Kuang, Xiongwei Han, Tao Zhong, Jianfeng Feng, and Wenlian Lu. Automated optimization modeling via a localizable error-driven perspective. *arXiv preprint arXiv:2602.11164*, 2026.

Fanbin Lu, Zhisheng Zhong, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Arpo: End-to-end policy optimization for gui agents with experience replay. *arXiv preprint arXiv:2505.16282*, 2025.

Meta AI. Introducing meta Llama 3: The most capable openly available LLM to date, April 2024. URL https://ai.meta.com/blog/meta-llama-3/. Accessed: 2024-04-18.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

OpenAI. GPT4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

Ruotian Peng, Yi Ren, Zhouliang Yu, Weiyang Liu, and Yandong Wen. Simko: Simple pass@ k policy optimization. *arXiv preprint arXiv:2510.14807*, 2025.

Mohammadreza Pourreza, Hailong Li, Ruoxi Sun, Yeounoh Chung, Shayan Talaei, Gaurav Tarlok Kakkar, Yu Gan, Amin Saberi, Fatma Ozcan, and Sercan O. Arik. CHASE-SQL: Multi-path reasoning and preference optimized candidate selection in text-to-sql. *arXiv preprint arXiv:2410.01943*, 2024. URL https://arxiv.org/abs/2410.01943.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Y Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yuqiao Tan, Minzheng Wang, Shizhu He, Huanxuan Liao, Chengfeng Zhao, Qiunan Lu, Tian Liang, Jun Zhao, and Kang Liu. Bottom-up policy optimization: Your language model policy secretly contains internal policies. *arXiv preprint arXiv:2512.19673*, 2025. URL https://arxiv.org/abs/2512.19673.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, Haiqing Guo, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haotian Yao, Haotian Zhao, Haoyu Lu, Haoze Li, Haozhen Yu, Hongcheng Gao, Huabin Zheng, Huan Yuan, Jia Chen, Jianhang Guo, Jianlin Su, Jianzhou Wang, Jie Zhao, Jin Zhang, Jingyuan Liu, Junjie Yan, Junyan Wu, Lidong Shi, Ling Ye, Longhui Yu, Mengnan Dong, Neo Zhang, Ningchen Ma, Qiwei Pan, Qucheng Gong, Shaowei Liu, Shengling Ma, Shupeng Wei, Sihan Cao, Siying Huang, Tao Jiang, Weihao Gao, Weimin Xiong, Weiran He, Weixiao Huang, Weixin Xu, Wenhao Wu, Wenyang He, Xianghui Wei, Xianqing Jia, Xingzhe Wu, Xinran Xu, Xinxing Zu, Xinyu Zhou, Xuehai Pan, Y. Charles, Yang Li, Yangyang Hu, Yangyang Liu, Yanru Chen, Yejie Wang, Yibo Liu, Yidao Qin, Yifeng Liu, Ying Yang, Yiping Bao, Yulun Du, Yuxin Wu, Yuzhi Wang, Zaida Zhou, Zhaoji Wang, Zhaowei Li, Zhen Zhu, Zheng Zhang, Zhexu Wang, Zhilin Yang, Zhiqi Huang, Zihao Huang, Ziyao Xu, Zonghan Yang, and Zongyu Lin. Kimi k1.5: Scaling reinforcement learning with llms, 2025. URL https://arxiv.org/abs/2501.12599.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, 2025.

Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints. *ArXiv*, abs/2309.16240, 2023. URL https://api.semanticscholar.org/CorpusID:263142109.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning, 2025. URL https://arxiv.org/abs/2506.01939.

Wei Xiong, Chenlu Ye, Baohao Liao, Hanze Dong, Xinxing Xu, Christof Monz, Jiang Bian, Nan Jiang, and Tong Zhang. Reinforce-ada: An adaptive sampling framework under non-linear rl objectives, 2025. URL https://arxiv.org/abs/2510.04996.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, 2018.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=upehLVgq1b.

Runzhe Zhan, Yafu Li, Zhi Wang, Xiaoye Qu, Dongrui Liu, Jing Shao, Derek F Wong, and Yu Cheng. Exgrpo: Learning to reason from experience. *arXiv preprint arXiv:2510.02245*, 2025.

Hongzhi Zhang, Jia Fu, Jingyuan Zhang, Kai Fu, Qi Wang, Fuzheng Zhang, and Guorui Zhou. Rlep: Reinforcement learning with experience replay for llm reasoning. *arXiv preprint arXiv:2507.07451*, 2025.

Yixiao Zhou, Yang Li, Dongzhou Cheng, Hehe Fan, and Yu Cheng. Look inward to explore outward: Learning temperature policy from llm internal states via hierarchical rl. *arXiv preprint arXiv:2602.13035*, 2026.

Chunzheng Zhu, Yangfang Lin, Shen Chen, Yijun Wang, and Jianxin Lin. Medeyes: Learning dynamic visual focus for medical progressive diagnosis. *arXiv preprint arXiv:2511.22018*, 2025a.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025b.

A Limitations and Broader Impacts

Scope and uncertainty. The experiments cover dense language models from 1.7B to 8B parameters and the specified mathematics, SQL, and coding tasks. They do not establish behavior at substantially larger scales or in settings without reliable verifiable rewards. Three independent runs are available for GRPO, DyJR, RLEP, and Ex-GRPO in the main mathematical comparison, but most auxiliary studies and cross-domain comparisons are single runs. Small performance gaps in those studies should not be interpreted as statistically established differences.

Mechanism and measurement. Confidence-based admission and JS utilization have complementary effects. The matched controls do not support attributing all improvement to JS alone. Recency is a proxy for reference relevance, and reward filtering deliberately biases replay toward successful responses. The sampled regularizer is not an exact divergence to an explicitly evaluated mixture. Token statistics and Pass@k are indirect evidence of plasticity; the controlled SQL protocols and LLM-judged mathematical modes address different aspects of diversity. Their finite sample budgets, judge dependence, and inclusion of coherent incorrect strategies prevent conclusions about complete preservation of useful semantic alternatives or a universal causal mechanism.

Selection and hyperparameters. The main comparison uses the common step-240 endpoint. Checkpoints were inspected on the evaluation benchmarks without an independent validation split; although the selected endpoint is the same for every compared method and run, this is a limitation of the evaluation protocol. The warm-up boundary of 20 steps was chosen from the observed early transition and is setting-dependent. A matched 10/20/40-step study was not completed, and the available coefficient comparisons are not exhaustive. For a new setting, a short pilot can identify when smoothed loss or entropy leaves the initial rapid-change regime. Some auxiliary experiment details, including the entropy-control coefficient and complete decoding settings for the semantic audits, require more complete archival documentation for exact replication.

Efficiency. The buffer's steady-state capacity is smaller than its warm-up capacity. The isolated memory benchmark excludes cached log-probabilities and training state, fills the specified replay capacities, and sums allocations over four devices. Its scaling result should not be equated with complete training memory or the typical occupancy of every replay implementation. End-to-end runtime depends on hardware, sequence lengths, and batching.

Broader impacts. More efficient RLVR can reduce the resources needed to adapt reasoning models, while better preservation of alternative solutions may benefit applications with verifiable outcomes. Verification remains task-dependent: a rewarded answer need not imply a faithful reasoning process, and generated code or SQL can still be incorrect or unsuitable for deployment. Stronger generation capabilities can also be misused. The experiments do not establish safety, privacy, fairness, or reliability in deployed systems; independent testing and appropriate execution controls remain necessary in such uses.

B Additional Method Details

B.1 GRPO Background

For a prompt x, GRPO samples G responses from the rollout policy $\pi_{\theta_{\text{old}}}$. Let μ_x and σ_x be the group reward mean and standard deviation. The response-level advantage is the group-normalized reward, $\widehat{A}_i = (r(x, y_i) - \mu_x)/(\sigma_x + \varepsilon)$, with a numerical stabilizer ε. The same advantage applies to the response's tokens. At position j, the policy ratio is

$$\rho_{i,j}(\theta) = \frac{\pi_\theta(y_i^{(j)} \mid x, y_i^{(<j)})}{\pi_{\theta_{\text{old}}}(y_i^{(j)} \mid x, y_i^{(<j)})}. \tag{9}$$

The standard clipped per-token surrogate is

$$\ell_{i,j}^{\text{GRPO}} = -\min\Big[\rho_{i,j}\widehat{A}_i,\ \text{clip}(\rho_{i,j}, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}})\widehat{A}_i\Big]. \tag{10}$$

The training configuration uses token-mean aggregation, $\epsilon_{\text{low}} = 0.2$, and $\epsilon_{\text{high}} = 0.28$. DyJR adds its replay loss to this online objective. Replay rows do not receive an online GRPO advantage or an off-policy reward-gradient term.

B.2 Divergence Background and Surrogate Normalization

For distributions P and Q, let $R = (P + Q)/2$. Then

$$D_{\text{KL}}(P\|Q) = \mathbb{E}_{z\sim P} \log \frac{P(z)}{Q(z)}, \tag{11}$$

$$D_{\text{JS}}(P\|Q) = \tfrac{1}{2} D_{\text{KL}}(P\|R) + \tfrac{1}{2} D_{\text{KL}}(Q\|R). \tag{12}$$

Conventional JS is symmetric and bounded between zero and $\ln 2$. Where the density ratio $u = P/Q$ is defined, direct substitution gives

$$2D_{\text{JS}}(P\|Q) = \mathbb{E}_{z\sim Q}\left[u \log u - (u+1) \log \frac{u+1}{2}\right]. \tag{13}$$

This explains the factor of two in the implemented generator and its absorption into α_{JS}. The identity assumes exact sampling from Q. DyJR instead uses success-filtered historical trajectories and each trajectory's cached generating-policy probability. The selected historical sample distribution is not the unconditioned generating policy, and its denominator is not an explicitly evaluated mixture density. The resulting token-level sample objective is a JS-motivated surrogate.

Bounded divergence is a property of the full distributional expectation, not an upper bound on individual values of $f(u)$ or their gradients. Likewise, the GRPO clipped surrogate is not a hard constraint on policy distance. Our discussion of a recent policy neighborhood is a motivation tested by recency ablations, not a theoretical guarantee. No trajectory-level importance correction is claimed for this regularization reference.

B.3 DyJR Algorithm

Algorithm 1 Dynamic Jensen-Shannon Replay

Input: Dataset $\mathcal{D}$, policy π_θ, age limit $M = 8$, coefficient $\alpha_{\text{JS}} = 0.05$.
Initialize replay buffer $\mathcal{S} \leftarrow \varnothing$.
for training step $t = 1, \ldots, T$ **do**
  Freeze rollout parameters $\theta_{\text{old}} \leftarrow \theta$.
  Generate $G = 8$ responses for each of 512 prompts; cache sampled-token log-probabilities.
  Compute rewards, group advantages, and prompt confidences $C(x)$.
  Evict entries generated more than M steps ago; current entries have not yet been inserted.
  If $|\mathcal{S}| \geq 512$, draw 512 historical rows uniformly without replacement once; otherwise skip replay.
  Combine online and replay rows with source-type tags and shuffle the pool.
  Make one optimizer epoch with mini-batch size 128: GRPO on online rows; JS on replay rows.
  Keep the selected replay batch fixed throughout the optimizer epoch.
  Set admission rate to 20% for the first 20 steps and 5% thereafter.
  Select successful current trajectories by descending $C(x)$, subject to the admission budget.
  After optimization, insert selected trajectories with their original rollout-time log-probabilities and timestamp t.
end for

With the full replay batch, the 4,608-row pool yields 36 optimizer steps. This is one pass over the joint pool, not 36 resamplings of replay or repeated epochs. The replay-to-online trajectory ratio is 1:8, and replay occupies one ninth of the combined pool. Individual mini-batches need not preserve that ratio.

C Additional Related Work

Reinforcement Learning with Verifiable Rewards RLVR [Yue et al., 2025, Lambert et al., 2025] has emerged as a cornerstone for enhancing the reasoning capabilities of LLMs in domains with objective ground truth, such as mathematics and coding [Shao et al., 2024, Guo et al., 2025, Team, 2024, Li et al., 2024c]. Following the paradigm established by models like OpenAI o1 [OpenAI, 2024] and DeepSeek-R1 [Guo et al., 2025], recent efforts such as QwQ [Team, 2025] and Kimi k1.5 [Team et al., 2025] have leveraged verifiable feedback to scale test-time compute. While effective, standard RLVR typically relies on on-policy optimization, which often suffers from sample inefficiency and training instability when dealing with long-sequence reasoning trajectories.

D Experiment Setups

Table 7: Experimental settings adopted in this work. We employ the AdamW optimizer with $\beta_1 = 0.9, \beta_2 = 0.95$ and a weight decay of 0.1. The gradient clipping threshold is set to 1.0 for all models.

Category	Parameters	GRPO	DAPO	DPH-RL	DyJR	RLEP
Inference	G (Group Size)	8	8	8	8	8
	Prompt Batch Size	512	512	512	512	512
	Temperature	1.0	1.0	1.0	1.0	1.0
	Top-p / Top-k	1.0 / -1	1.0 / -1	1.0 / -1	1.0 / -1	1.0 / -1
	Evaluation Temp	0.7	0.7	0.7	0.7	0.7
	Evaluation Top-p	0.95	0.95	0.95	0.95	0.95
	Response Length	6144	6144	6144	6144	6144
Training	Clip Ratio High (ϵ)	0.28	0.28	0.28	0.28	0.28
	Clip Ratio Low (ϵ)	0.2	0.2	0.2	0.2	0.2
	Learning Rate	2×10^{-5}	2×10^{-5}	2×10^{-5}	2×10^{-5}	2×10^{-5}
	Mini Batch Size	128	128	128	128	128
	Online Loss Agg.	token-mean	token-mean	token-mean	token-mean	token-mean
	Dynamic Sampling	False	True	True	False	False
	Replay Bsz	-	-	512	512	512
	Correct Threshold	-	-	6 of 8	-	-
	Max Age (M)	-	-	-	8	-
	Regularization Coefficient	-	-	0.2	0.05	-
Hardware	GPU Count	32 × A800	32 × A800	32 × A800	32 × A800	32 × A800
	Parallel (TP/SP)	1 / 1	1 / 1	1 / 1	1 / 1	1 / 1

We evaluate the following tasks: (1) A **long-chain mathematical reasoning task** based on the Qwen3-4B-Base [Yang et al., 2025] model. For training, we utilize Reinforce-Ada-Hard [Xiong et al., 2025], a high-difficulty subset of 14k samples filtered from the OpenR1-Math-220k[1] dataset, specifically excluding simpler prompts with an average success rate (average reward $>$ 0.375). Evaluation is performed on highly challenging benchmarks, including AIME25 [Li et al., 2024a], HMMT25 [Balunović et al., 2025], BRUMO25 [Balunović et al., 2025], AMC23, Minerva [Lewkowycz et al., 2022] and Beyond AIME. This task is designed to observe the significant performance leap from a low baseline to high-level proficiency following RL training. (2) A **SQL generation task** based on Llama-3.1-8B-Instruct [Meta AI, 2024], which focuses on moderate-difficulty reasoning typically within a 1k token limit. The model is trained on the BIRD [Li et al., 2024b] dataset and evaluated across both BIRD and Spider [Yu et al., 2018] datasets to test cross-domain performance.

For coding, we train Qwen2.5-7B-Instruct on CodeContests problems of difficulty at most 2 and evaluate LiveCodeBench-V2 and a held-out CodeContests split, reporting Pass@16 under unit-test verification.

In this work, we employ multiple experimental setups to validate the effectiveness of **DyJR** in comparison with baseline methods; Table 7 summarizes the main mathematical training configuration. SQL uses the shorter responses described above; the isolated storage study has its own hardware and length settings in Appendix H.6. Specifically, *Replay Bsz* denotes the historical replay batch used in an update, not the number of newly admitted trajectories. New admission is controlled separately

[1] https://huggingface.co/datasets/open-r1/OpenR1-Math-220k

Table 8: ExGRPO-specific hyperparameters for experience replay.

Parameter	Value	Description
Replay ratio (ρ)	0.5	Fraction of replay samples in each mini-batch
Delayed replay threshold	0.35	Minimum batch Pass@1 required to activate experience replay
Difficulty bucket range	$[0, 7]$	Bounds for difficulty-based bucketing (k/K)
Replay metric	Entropy	Criterion for evaluating trajectory quality
Trajectory selection mode	Argmin	Select the lowest-entropy successful trajectory per prompt

by the 20%/5% schedule. For **DPH-RL**, following the experimental protocol in their original work, we incorporate samples that yield 6 correct responses out of 8 sampled paths into the regularization term. Regarding **RLEP**, we admit two correct solution trajectories per query per update, with up to 64 retained trajectories per query. These buffered trajectories participate in the replay baseline's policy updates, in contrast with DyJR's regularization-only utilization. For Pass@1, we employ greedy decoding (i.e., temperature $T = 0$). For Mean@256, we generate 256 samples with a temperature of $T = 0.7$ and calculate the average accuracy across all samples.

ExGRPO. (Experiential Group Relative Policy Optimization) [Zhan et al., 2025] extends GRPO with an experience replay framework that integrates *experience management* and *mixed optimization*. During training, ExGRPO maintains an experience pool that maps each prompt to a set of previously successful solution trajectories, and categorizes prompts into difficulty buckets according to their most recent rollout accuracy. To prevent overfitting to trivial cases, prompts whose rollouts are entirely correct are moved into a *retired set* and excluded from subsequent replay.

At each update step, ExGRPO applies a replay ratio ρ to partition each mini-batch into on-policy samples and replayed experiences. Difficulty buckets are sampled using a Gaussian weighting centered at 0.5, thereby emphasizing prompts of intermediate difficulty. For each selected prompt, ExGRPO further selects the successful trajectory with the lowest entropy under the current policy, encouraging stable exploitation while mitigating excessive plasticity collapse.

The additional hyperparameters introduced by ExGRPO are summarized in Table 8, while all other training configurations follow the GRPO baseline in the Table 7.

Training budget and evaluation. Main long-chain runs use 32 A800 GPUs and take approximately 20 hours each, or approximately 640 GPU-hours per run. Checkpoints are evaluated every 80 steps, and step 240 is the reported endpoint for every method and run in the main comparison. No separate validation set is used. The reported runtime does not provide a complete aggregate accounting of all pilots, auxiliary studies, or failed runs.

E Admission Schedule and Replay Controls

Table 9: Performance of DyJR on various mathematical benchmarks. $init$ denotes the percentage of rollout samples retained per step during the initial 20 steps, while η represents the retention rate thereafter. The maximum age of the FIFO buffer is consistently set to 8 steps.

Model	AIME25	AMC23	Beyond AIME	BRUMO25	HMMT25	Minerva*	Avg
DyJR ($init = 20\%, \eta = 20\%$)	22.9	**72.5**	12.5	**33.8**	12.6	**51.0**	**34.2**
DyJR ($init = 20\%, \eta = 5\%$)	**23.1**	72.0	**12.7**	33.4	**12.7**	50.4	34.1
DyJR ($init = 10\%, \eta = 5\%$)	22.5	69.2	11.4	31.5	11.5	48.2	32.4
DyJR ($init = 5\%, \eta = 5\%$)	21.8	68.5	10.1	29.2	10.3	47.8	31.3

Table 9 compares admission rates under the same age constraint. Constant 20% admission and the 20%→5% schedule obtain 34.2 and 34.1. The result supports storage-efficient contraction after adequate early population, without establishing that late trajectories are intrinsically less valuable. Lower initial admission yields 32.4 at 10% and 31.3 at 5%. These single-run comparisons do not supply confidence intervals for the small 34.2–34.1 difference.

F Replay Selection and Sparse Rewards

Filtering replay samples by reward introduces selection bias. DyJR does not aim to estimate the full rollout distribution unbiasedly; instead, it constructs a high-quality reference for plasticity regularization. A stricter rule that admits successful trajectories only from fully correct prompt groups can cause buffer starvation in harder settings. The confidence-stratified admission rule improves the tradeoff between replay quality and replay availability.

Table 10 compares perfect-only replay with confidence-based replay. Increasing M from 8 to 32 raises the perfect-only result from 30.4 to 31.3, but reduces the confidence-based result from 34.1 to 31.7. More storage can alleviate scarcity under a restrictive rule, whereas retaining older samples can hurt when fresh anchors are already available. The matched admission–objective comparison appears in Table 2.

Table 10: Replay selection and staleness. Confidence-based selection reduces buffer starvation, while overly stale replay remains harmful.

Replay Selection	**Max Age M**	**Avg**
Perfect-only	8	30.4
Perfect-only	32	31.3
Confidence-based	8	**34.1**
Confidence-based	32	31.7

We also evaluate a harder sparse-reward setting with Qwen3-1.7B-Base, where the initial reward is much lower and final reward remains modest. Table 11 shows that DyJR still improves over GRPO, suggesting that the method does not rely on a particularly strong initial policy.

Table 11: Sparse-reward setting with Qwen3-1.7B-Base.

Method	**Avg**
GRPO (1.7B, sparse)	17.3
DyJR (1.7B, sparse)	**19.5**

G Alternative Regularizers and Off-Policy Corrections

We further compare DyJR with common alternatives under the same compute budget. The evaluated alternatives include static-reference KL, a dynamic entropy bonus, and direct off-policy replay with importance ratios. These single-run results compare the specified configurations rather than all possible variants of each regularizer. Table 12 summarizes the results.

Table 12: Comparison with alternative regularization and off-policy correction strategies under the same compute budget.

Method	**Avg**
KL-to-ref (static reference)	30.8
Dynamic entropy bonus	30.3
Replay + off-policy PG w/ IS ratio	32.6
DyJR	**34.1**

H Additional Evaluation of Replay, Diversity, and Memory

This section evaluates the robustness of replay-baseline comparisons, separates replay admission from replay utilization, and examines diversity beyond token-level summaries. We also report an isolated storage benchmark. These experiments assess complementary aspects of DyJR; none alone establishes a complete causal account of local policy plasticity.

H.1 Repeated Runs of Replay Baselines

We evaluate Ex-GRPO and RLEP with three independent runs each, including the originally reported run, in the same VERL training stack used for the main comparisons. GRPO and DyJR also use three independent runs. Method-specific hyperparameters follow the respective implementations. Checkpoints are evaluated every 80 training steps on the same six benchmarks, without a separate validation set. For every method and run in this comparison, the highest six-benchmark average occurs at the final evaluated checkpoint, step 240. Thus, the reported comparison is identical to reporting step 240 uniformly, although benchmark-based checkpoint selection can introduce optimistic bias in general.

Table 13 reports the means and sample standard deviations for all four repeated-run methods; Table 14 gives the individual Ex-GRPO and RLEP runs. Ex-GRPO obtains 32.6 and RLEP obtains 31.8 on the six-benchmark average, compared with 34.1 for DyJR. The observed mean differences are 1.5 and 2.3 percentage points, respectively. We report these as repeated-run comparisons rather than formal significance tests; three runs provide only a limited estimate of training variability.

Table 13: Variability across three independent training runs. Entries are reported means $\pm$ sample standard deviations, in percentage points. Minerva uses Mean@16; the other benchmarks use Mean@256. Avg is the six-benchmark average. The aggregate standard deviation for GRPO was not reported, so only its mean is shown.

Benchmark	GRPO	RLEP	Ex-GRPO	DyJR
AIME25	19.1 ± 1.8	22.6 ± 0.5	22.2 ± 0.4	23.1 ± 1.6
AMC23	64.6 ± 1.5	70.9 ± 1.7	69.9 ± 0.7	72.0 ± 1.1
Beyond AIME	10.7 ± 0.9	10.9 ± 0.3	11.4 ± 1.7	12.7 ± 0.7
BRUMO25	29.6 ± 1.6	28.7 ± 0.8	33.0 ± 0.2	33.4 ± 1.4
HMMT25	9.8 ± 0.8	10.7 ± 1.6	10.0 ± 1.3	12.7 ± 0.8
Minerva	45.2 ± 1.4	47.3 ± 1.5	49.2 ± 0.9	50.4 ± 2.6
Avg	29.8	31.8 ± 0.2	32.6 ± 0.2	**34.1** ± 0.3

Table 14: Individual runs of replay baselines. Values are percentages; the final column is the average over six benchmarks. Run 1 is the originally reported run. Reported rounded values are preserved; their three-run summaries appear in Table 13.

Method	Run	AIME25	AMC23	Beyond AIME	BRUMO25	HMMT25	Minerva*	Avg
Ex-GRPO	1	21.7	70.2	11.6	32.8	10.4	50.1	32.8
	2	22.5	70.3	12.9	33.0	8.5	48.4	32.6
	3	22.4	69.1	9.6	33.2	11.1	49.0	32.4
RLEP	1	23.0	68.9	10.6	28.9	10.8	48.4	31.8
	2	22.7	72.2	11.1	29.3	9.1	47.8	32.0
	3	22.0	71.5	11.1	27.8	12.3	45.6	31.7

H.2 Separating Replay Admission from Utilization

Confidence-based admission changes which trajectories enter the buffer, whereas the replay objective determines how the available trajectories affect optimization. To separate these factors, we compare perfect-only and confidence-based admission under both JS regularization and off-policy policy-gradient replay with importance sampling (PG+IS). All four settings use $M = 8$ and matched buffer configuration and training budget. The confidence-based PG+IS setting uses the same default admission rule as DyJR.

Changing from perfect-only to confidence-based admission improves the average from 30.4 to 34.1 with JS and from 30.1 to 32.6 with PG+IS. This is consistent with the importance of avoiding buffer starvation when fully correct trajectories are sparse. Once confidence-based admission is fixed, JS

Table 15: Matched admission–utilization comparison. All settings use $M = 8$ and the same replay-buffer configuration and training budget. Avg is the six-benchmark average accuracy (%).

Admission	Replay utilization	Avg
Perfect-only	JS regularization	30.4
Confidence-based	JS regularization (DyJR)	**34.1**
Perfect-only	Off-policy PG+IS	30.1
Confidence-based	Off-policy PG+IS	32.6

improves the average by a further 1.5 points over PG+IS. These results support complementary roles: admission provides sufficient fresh anchors, and the JS objective makes more effective use of those anchors in this setting. They do not attribute the entire gain over GRPO to JS alone.

H.3 Standard Token-Level Entropy Regularization

We add a standard token-level entropy bonus to GRPO under the same training setup:

$$\mathcal{L}_{\text{entropy}} = \mathcal{L}_{\text{GRPO}} - \beta\, \mathbb{E}_t[H(\pi_\theta(\cdot \mid s_t))]\,, \tag{14}$$

where β is the entropy coefficient. Table 16 shows essentially unchanged average accuracy for this configuration: 29.7, compared with 29.8 for GRPO and 34.1 for DyJR. The entropy bonus encourages uncertainty across response-token distributions, whereas DyJR conditions its preservation signal on recently accepted replay trajectories. This comparison supports the value of a targeted replay anchor in the evaluated configuration; it is not an exhaustive study of entropy coefficients or schedules.

Table 16: Standard token-level entropy regularization. Avg is the six-benchmark average accuracy (%).

Method	Avg
GRPO	29.8
GRPO + standard entropy regularization	29.7
DyJR	**34.1**

H.4 Controlled Protocol Diversity in NL2SQL

Token-level entropy cannot distinguish alternative reasoning protocols from lexical variation. We therefore conduct a controlled NL2SQL experiment with five pre-constructed protocol families used for supervised fine-tuning. Following the multi-path construction in CHASE-SQL [Pourreza et al., 2024], these comprise divide-and-conquer reasoning, query-plan-first reasoning, direct step-by-step reasoning, and trajectories distilled from DeepSeek-R1 and GPT-4o. Each family has a distinct identifying prefix (Table 17); the prefix is a label for the constructed family, rather than an independent definition of semantic difference.

Table 17: Five protocol families used in the controlled NL2SQL study. Distinct prefixes provide deterministic family labels.

Protocol family	Identifying response prefix
Divide-and-conquer	"Divide and Conquer ..."
Query-plan-first	"Query Plan ..."
Direct step-by-step	"Let's break down and think step by step ..."
DeepSeek-R1-distilled	"Okay, to answer the question ..."
GPT-4o-distilled	"The original question can ..."

For each test question, we draw 32 responses and count the distinct protocol labels observed. Before RL, approximately 80% of questions exhibit at least five labels. After GRPO, 95% exhibit only one. After DyJR, 32% exhibit two, 30% three, and 27% four, totaling 89% with at least two observed protocol families (Table 18). This controlled result indicates that DyJR maintains a broader set

of constructed solution protocols under the same sample budget. It does not establish that every retained response is correct, or that a protocol prefix by itself verifies the semantics of an unconstrained derivation.

Table 18: Observed protocol diversity in NL2SQL with 32 samples per question. Percentages describe questions, not individual responses.

Model	Observed protocol counts
SFT base	Approximately 80% of questions exhibit at least five protocols.
GRPO	95% of questions exhibit one protocol.
DyJR	32% exhibit two; 30% three; 27% four; 89% exhibit at least two.

H.5 Mathematical Strategies from Complete Derivations

To examine mathematical reasoning beyond prefixes, we fix 50 problems and sample 16 responses per problem from each of Qwen3-4B-Base, GRPO, and DyJR, giving 800 responses per model. For each model–problem pair, GPT-5.6-sol receives the problem, all 16 complete responses, and fixed correctness labels. The judge identifies the mathematical method executed by each response and clusters responses that share the same core strategy.

Coherent but incorrect attempts are retained. Differences restricted to wording, calculation order, arithmetic or algebraic errors, or minor reparameterizations are merged when the underlying strategy is unchanged. Responses without an identifiable strategy are excluded. Consequently, the reported modes describe identifiable mathematical strategies and are not restricted to strategies containing a correct response.

DyJR exhibits at least two modes on 58% of problems, compared with 12% for GRPO, and its mean mode count is 2.2 rather than 1.3. Both measures remain below the base model's 78% and 2.6, respectively. Thus, the result supports greater observed strategy diversity after DyJR than after GRPO on this fixed problem set; it does not imply complete preservation of the base model's alternatives.

For example, AMC 2023 Problem 17 asks for the number of digits in $8^5 \cdot 5^{10} \cdot 15^5$. The base model exhibits both exact prime-factor pairing, yielding $243 \cdot 10^{15}$ before counting digits, and the logarithmic rule $\lfloor \log_{10} N \rfloor + 1$. On AIME 2024 item 24 in the evaluation set, the identified alternatives include logarithmic linearization followed by solving a linear system, and direct substitution after converting the equations to exponential form.

This analysis uses an LLM judge and a small fixed set of problems. Clustering can be sensitive to judge decisions and sampling coverage, and diversity among all identifiable strategies differs from diversity among correct solutions. Moreover, comparing GRPO with the complete DyJR method does not isolate the contribution of JS from that of admission and buffer construction. We therefore treat these results as complementary empirical evidence for the plasticity interpretation, alongside the matched objective ablation and sampling-budget analyses.

H.6 Isolated Replay-Storage Scaling

We isolate storage scaling by materializing replay buffers on four NVIDIA H100 80GB GPUs. Every trajectory contains 8,192 tokens represented solely by `int32` token IDs; stored log probabilities, masks, other metadata, model parameters, and optimizer states are excluded. RLEP is evaluated at its specified maximum capacity of 64 trajectories per training query. Although it collects two trajectories per query at an update, that collection rate is distinct from its maximum retained capacity. The DyJR scenario fixes the total capacity to 1,638 trajectories, independently of the number of training queries Q.

The metric is the net increase in resident GPU memory reported by `nvidia-smi`, summed across the four devices while the complete buffers are live. It includes allocation/runtime overhead beyond the tensor payload, and is not the total memory consumed by a training run. For example, the DyJR token-ID tensors occupy approximately 0.050 GiB, whereas the measured resident-memory increase is 0.41 GiB.

The measured RLEP footprint increases with the number of queries because the benchmark fills a separate buffer for each query. The fixed-capacity DyJR footprint remains constant across the tested query counts. These measurements establish the storage-scaling difference under the stated capacity and representation choices; they do not imply that every RLEP run reaches full capacity or that a complete DyJR replay cache contains only token IDs.

H.7 Interpretation of Post-Warm-Up Contraction

Constant 20% admission obtains an average of 34.2, compared with 34.1 for the $20\% \to 5\%$ schedule. This comparison supports reducing replay admission and capacity after warm-up without an observed performance cost in this setting. It does not establish that later trajectories are intrinsically less useful. Conversely, the reduced-initial-admission ablations indicate the importance of adequate early buffer population. The evidence therefore favors sufficient early anchor availability followed by storage-efficient contraction, rather than a general ranking of early versus late data quality.

I Additional Sampling-Budget and Timing Analysis

We evaluate Beyond AIME as the sampling budget increases to 1,024 responses per problem. The Pass@k curves in Figure 4 place JS-based DyJR above GRPO and the forward-KL variant throughout the reported range. The rollout and actor-update panels show similar measured latency for GRPO and DyJR in this configuration; they are not a universal bound on runtime overhead. Actor updates include log-probability calculation, forward passes, and gradient updates. The replay regularizer is $\mathcal{L}_{\mathrm{JS}}$, not an SFT objective.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction describe the empirical scope and the complementary admission and utilization findings; limitations of the mechanism evidence are discussed in Section 5 and Appendix A.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: Appendix A discusses model scale, sampling and judge limitations, checkpoint selection, uncertainty, hyperparameters, and efficiency measurement.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A].

Justification: The paper is empirical and does not claim new formal theoretical guarantees. Standard objectives and the algebraic normalization of the regularizer are specified in Section 3 and Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [No].

Justification: The main training configuration, algorithm, update schedule, evaluation metrics, and checkpoint protocol are supplied. Exact replication of some auxiliary semantic and entropy studies still requires parameters and archived records noted in Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No].

   Justification: The source datasets and model families are identified, but this version does not supply a verified public URL for the exact training implementation and complete auxiliary-study records.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [No].

   Justification: Section 4 and Appendix D specify the main settings. Some auxiliary-study details, including the entropy coefficient and complete semantic-audit decoding settings, are not fully documented.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: Three independent runs are reported for GRPO, DyJR, RLEP, and Ex-GRPO in the main mathematical comparison. Table 13 in the appendix reports sample standard deviations across runs; single-run studies are explicitly identified, and no formal significance test is claimed.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No].

   Justification: Appendix D reports the main hardware and approximate per-run cost, and Appendix H.6 specifies the isolated memory benchmark. A complete total-project accounting, including pilots and failed runs, is unavailable.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: The work uses existing research datasets and models, reports empirical limitations, and includes no human-subject study. The described research does not require a stated deviation from the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes].

Justification: Appendix A discusses resource-efficiency benefits, verification and reliability limits, and the need to assess risks before deployment.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A].

Justification: This manuscript does not accompany a new high-risk model or scraped-dataset release requiring additional release safeguards.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [No].

Justification: Model and dataset sources are credited in Section 4 and Appendix D. Version-specific licenses and terms are not fully documented for all assets.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A].

    Justification: This manuscript does not introduce a new dataset or model release. The method and experiments are described in the paper and appendices.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: The experiments do not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A].

Justification: The experiments do not involve human subjects, so an IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes].

Justification: Appendices H.4 and H.5 identify the DeepSeek-R1/GPT-4o-distilled SQL protocols and GPT-5.6-sol mathematical-strategy judge. These empirical roles go beyond writing assistance.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.