Advancing Entropy-Level Credit Assignment in RLVR via Proximal Entropy Policy Optimization

Yun Kim
Seoul National University
SK Hynix
yunkimmy@snu.ac.kr

Nojun Kwak
Seoul National University
nojunk@snu.ac.kr

Abstract

Value-model-free RLVR methods such as GRPO assign uniform advantages to all tokens in a rollout, ignoring that tokens contribute unequally. Recent methods use token entropy as an importance proxy but compute it globally across the batch, conflating importance with prompt difficulty and positional trends. We argue that importance should instead be measured relative to a token's own local context, and introduce proximal entropy: a local measure of token importance relative to neighboring tokens, and prove it is invariant to both confounders. Proximal Entropy Policy Optimization (PEPO) uses it to weight per-token advantages and outperforms GRPO and entropy-based baselines on mathematical reasoning across Qwen3-1.7B, Qwen3-4B, and Llama-3.2-3B-Instruct. We also show the formulation generalizes to other algorithms where substituting proximal entropy into existing methods improves, and applying it to single-stream RL succeeds where global entropy fails.

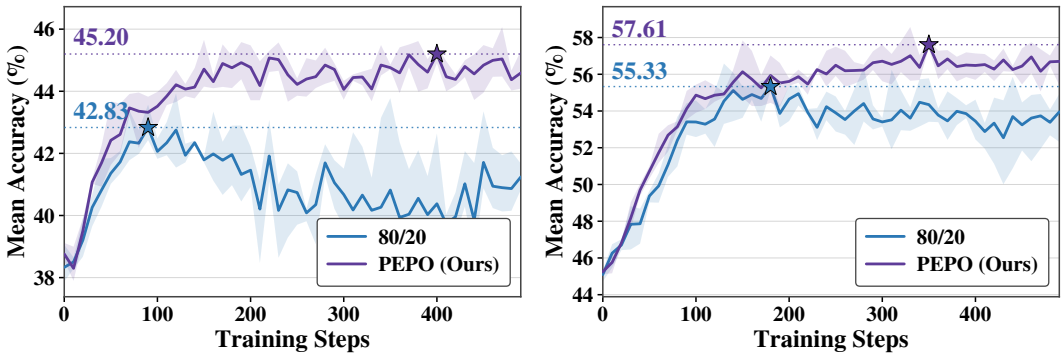
1 Introduction

In Reinforcement Learning with Verifiable Rewards (RLVR) Lambert et al. [2024], Guo et al. [2025a] for large language models, value-model-free methods such as Group Relative Policy Optimization (GRPO) Shao et al. [2024] have become dominant for their efficiency. However, these methods assign a single sequence-level advantage to every generated token, ignoring the fact that tokens contribute unequally to the final answer. Several approaches address this through learned value models Schulman et al. [2017], Parthasarathi et al. [2025] or additional rollouts Kazemnejad et al. [2025], Guo et al. [2025b], but these require training a separate model or generating extra samples, adding significant overhead. Among lighter alternatives, token entropy has emerged as the most widely adopted proxy for token importance Wang et al. [2025a], Cheng et al. [2025a], Tan et al. [2025], requiring no overhead beyond the standard forward pass.

The premise of entropy-based credit assignment is that low-entropy tokens correspond to routine grammatical continuations, while high-entropy tokens mark *pivotal decisions* in the reasoning trajectory Wang et al. [2025a]. Methods such as 80/20 Wang et al. [2025a] exploit this by updating only the top 20% of tokens by entropy, ranked across all tokens in a training batch. However, we observe that this batch-level approach systematically biases updates toward hard prompts and early generation steps, where entropy is globally elevated. This causes low-importance tokens in hard prompts or early positions to be prioritized over genuine decision points, misguiding the learning signal. Therefore, we argue that a token's importance should be measured relative to its own *local context*, not relative to the batch.

In this paper, we propose Proximal Entropy Policy Optimization (PEPO), which assigns each token an advantage weight based on its *proximal entropy*. Proximal entropy measures how uncertain the model is at a given position relative to its neighboring tokens, rather than relative to the entire batch. We define proximal entropy as a softmax over a sliding window of entropies centered on each

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



(a) Qwen3-1.7B (b) Qwen3-4B

Figure 1: Validation accuracy over training steps for PEPO and the 80/20. The plotted values represent the average mean@k accuracy evaluated across the MATH, AIME 2024, AIME 2025, and AMC benchmarks for Qwen3-1.7B (a) and Qwen3-4B (b). PEPO achieves a higher validation accuracy compared to the 80/20 baseline.

token, providing a per-token importance signal that is invariant to prompt difficulty and robust to positional trends. We formally show that this formulation removes both confounders (Proposition 1) and empirically verify that proximal entropy remains constant across difficulty levels and generation positions where global entropy varies substantially. Importantly, computing proximal entropy adds negligible overhead, as token-level entropy is precomputed by standard LLM inference engines and the sliding window operations are fully vectorized.

We conduct extensive experiments on MATH500 Lightman et al. [2024], AMC Mathematical Association of America [2023], and AIME 2024/2025 Mathematical Association of America [2024] using Qwen3-1.7B, Qwen3-4B Yang et al. [2025], and Llama-3.2-3B-Instruct Grattafiori et al. [2024], where PEPO consistently outperforms batch-level methods on Qwen3-1.7B and Qwen3-4B (Figure 1). Beyond accuracy gains, we find that the proximal entropy formulation is itself the key ingredient. Substituting our local reference frame into 80/20 improves its performance without any other changes, isolating the contribution from the rest of PEPO's design. The same formulation also transfers to single-stream RL (SPO) Xu and Ding [2025], where each prompt produces only a single rollout. In this setting, the diversity of prompts greatly increases, and batch-level entropy methods degrade below the baseline while PEPO continues to improve over it. Together, these results confirm that proximal entropy captures token importance more effectively than batch-level approaches, providing accurate credit assignment for value-model-free RL. We summarize our contributions as follows.

- We identify a systematic bias in batch-level entropy metrics Wang et al. [2025a], Cheng et al. [2025a] that concentrates updates on hard prompts and early generation steps.
- We propose PEPO, which weights per-token advantages using *proximal entropy*, a local measure of token importance that we formally prove is invariant to prompt difficulty and positional trends.
- We demonstrate that PEPO consistently outperforms batch-level methods across three model families (Qwen3-1.7B, Qwen3-4B, and Llama-3.2-3B-Instruct) on four benchmarks (MATH500, AMC, AIME 2024, and AIME 2025).
- We show that the proximal entropy formulation generalizes beyond PEPO, improving 80/20 when substituted in and transferring to single-stream RL where batch-level entropy fails.

2 Related Work

Value-model-free RLVR and credit assignment. GRPO [Shao et al., 2024] and its successors [Yu et al., 2025, Zheng et al., 2025, Zhao et al., 2025, Chen et al., 2025] eliminate the value network of PPO by estimating advantages through group-relative comparisons, and Single-stream Policy Optimization [Xu and Ding, 2025] further removes the group synchronization barrier. These methods differ in how they estimate and stabilize advantages but share a common limitation that every token

in a rollout receives identical credit. A range of approaches address this through auxiliary signals. Learned value networks [Schulman et al., 2017, Parthasarathi et al., 2025], Monte Carlo estimates from extra rollouts or tree-structured branches [Kazemnejad et al., 2025, Guo et al., 2025b, Tran et al., 2025, Wang et al., 2026, Li et al., 2026], process reward models that score each step [Cheng et al., 2025b, Xie et al., 2025b, Zou et al., 2025], attention maps from selected heads [Li et al., 2025, Jiao et al., 2026, Nie et al., 2026], and gradient-based attribution [Zhang et al., 2026] all recover stronger credit signals at the cost of additional models, extra rollouts, or extra computation per token. Token entropy, by contrast, is computed from the same forward pass as the policy itself and adds essentially zero overhead, making it the lightweight signal we focus on in this work.

Token-entropy-based credit assignment. Token entropy has emerged as the dominant lightweight proxy for token importance, since it requires no overhead beyond the standard forward pass. Wang et al. [2025a] show that high-entropy tokens correspond to forking points in reasoning trajectories and propose 80/20, which restricts updates to the top 20% of tokens by entropy. Cheng et al. [2025a] augment the advantage with a clipped, gradient-detached entropy term to amplify updates at high-entropy positions. GTPO [Tan et al., 2025] redistributes rewards by weighting each token by its entropy relative to other rollouts in its group, while UCAS [Xie et al., 2025a] modulates advantages using both response-level confidence and token-level certainty. ERPO [Yu et al., 2026] and EDGE-GRPO [Zhang et al., 2025] further explore entropy-aware gating and advantage diversity. A common thread across these methods is that entropy statistics are computed in a global reference frame, either across the entire batch or as absolute per-token quantities. As we demonstrate in Section 3.2, this global frame conflates token importance with prompt difficulty and positional trends, leading to systematic biases in which tokens receive credit.

Closest to our work, ARES [Chen et al., 2026] uses window-averaged entropy to shape per-token advantages in multimodal reasoning, adding a bonus on tokens whose window entropy exceeds a batch-derived threshold. PEPO differs in three respects. First, ARES uses a forward-looking window with arithmetic averaging to *smooth* entropy and thresholds it against a batch-global cutoff, whereas PEPO uses a centered window with softmax-relative weighting that measures each token *against* its local neighbors and is therefore invariant to the absolute scale. Second, the symmetric window is essential to our positional invariance result (Proposition 1), since the cancellation of linear positional trends in the proof relies on offsets ranging symmetrically around t; a forward-looking window would retain a residual trend bias. Third, we show that the local frame is a drop-in improvement that transfers across algorithms and to the single-stream setting where global entropy methods fail.

3 Background and Motivation

3.1 Preliminaries

We consider the RLVR setting [Lambert et al., 2024, Guo et al., 2025a], in which a policy π_θ generates a response $y = (y_1, \ldots, y_T)$ conditioned on a prompt x, and a verifier assigns a binary reward $r(x, y) \in \{0, 1\}$ indicating whether y is correct. The goal is to maximize the expected reward $\mathbb{E}_{x\sim p_{data}, y\sim\pi_\theta(\cdot|x)}[r(x, y)]$.

GRPO. For each prompt x, GRPO [Shao et al., 2024] samples a group of G rollouts $\{y_i\}_{i=1}^G$ from the current policy and computes a group-normalized advantage

$$A_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^G)}{\text{std}(\{r_j\}_{j=1}^G)}, \tag{1}$$

where $r_i = r(x, y_i)$. The same advantage A_i is assigned to every token $y_{i,t}$ in the rollout, and the policy is updated by maximizing

$$\mathcal{J}_{\text{group}}(\theta) = \frac{1}{\sum_{j=1}^G T_j} \sum_{i=1}^{G} \sum_{t=1}^{T_i} \min\big(\rho_{i,t}(\theta)A_i,\ \text{clip}(\rho_{i,t}(\theta), 1-\varepsilon_l, 1+\varepsilon_h)A_i\big), \tag{2}$$

where T_i is the number of tokens generated in the i-th rollout, $\rho_{i,t}(\theta) = \pi_\theta(y_{i,t} \mid x, y_{i,<t})/\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t})$ is the importance ratio and $\varepsilon_l, \varepsilon_h$ are the lower and upper clipping thresholds [Schulman et al., 2017, Yu et al., 2025].

Token entropy. The entropy of the policy at step t of rollout i is

$$H_{i,t} = -\sum_{v \in \mathcal{V}} \pi_\theta(v \mid x, y_{i,<t}) \log \pi_\theta(v \mid x, y_{i,<t}), \tag{3}$$

where $\mathcal{V}$ is the vocabulary.

Fine-grained credit assignment. Because A_i in Eq. (1) is constant across t, GRPO assigns identical credit to every token in a rollout, regardless of its role in producing r_i. We refer to the problem of assigning per-token weights $c_{i,t}$, such that the effective advantage $c_{i,t}A_i$ reflects each token's contribution, as *fine-grained credit assignment*. The challenge is that r_i provides no direct signal about which tokens mattered, so any solution must rely on a proxy for token importance. A learned value model can provide such a proxy [Schulman et al., 2017, Parthasarathi et al., 2025], but introduces substantial memory and training overhead and reintroduces the instability issues that motivated value-free RLVR in the first place [Shao et al., 2024, Yu et al., 2025]. In this work, we explore token entropy as the importance signal, since it is a lightweight proxy that introduces no additional overhead, as it is already computed during standard LLM inference [Kwon et al., 2023].

3.2 Entropy as a Token Importance Signal

Token entropy has emerged as a model-free alternative to learned value models [Wang et al., 2025a, Cheng et al., 2025a, Tan et al., 2025]. During generation, routine continuations such as grammatical placeholders and function words carry little uncertainty, while tokens at logical branch points, intermediate numerical results, and operator choices exhibit high entropy and frequently determine whether a derivation proceeds correctly [Wang et al., 2025a]. A growing line of work exploits this by using entropy to instantiate the per-token weight $c_{i,t}$ [Wang et al., 2025a, Cheng et al., 2025a, Xie et al., 2025a]. For instance, Wang et al. [2025a] show that high-entropy tokens correspond to *forking* points in the reasoning trajectory and propose 80/20, which updates only the top 20% of tokens by entropy:

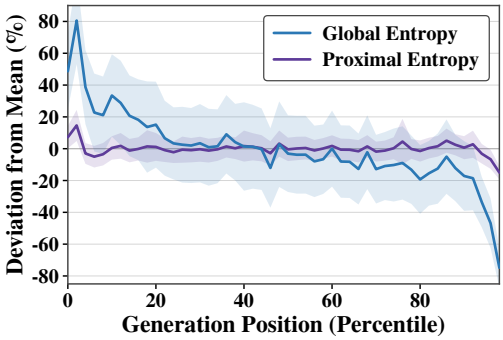


Figure 2: Positional bias in token entropy on Llama-3.2-3B-Instruct. We plot the deviation of global and proximal entropy from its sequence-level mean across generation percentiles.

$$c_{i,t} = \mathbb{1}\left[H_{i,t} \geq \tau_{80}\right], \tag{4}$$

where τ_{80} is the 80th-percentile entropy across all tokens in the batch. Cheng et al. [2025a] similarly augments the advantage function with a global entropy term to reinforce exploratory reasoning behaviors. In both cases, the reference frame is global.

However, a token's raw entropy is driven not only by its importance but also by two confounding factors: the token's position in the sequence and the difficulty of the prompt. Figure 2 illustrates the positional bias on Llama-3.2-3B-Instruct by plotting the deviation of global entropy from its sequence-level mean across generation percentiles, computed over 1024 rollouts from the validation set. Global entropy exhibits a strong positional trend, with early tokens deviating up to 80% above the mean and late tokens falling well below it, reflecting that early generation steps carry higher uncertainty before the model commits to a reasoning direction [Guo, 2025]. Table 1 illustrates the prompt difficulty bias. We group prompts by pass rate (out of 8) into hard (0–1), moderate (2–6), and easy (7–8) categories and report the mean entropy for each group. Global entropy varies substantially across difficulty levels (0.315 for hard vs. 0.221 for easy), confirming that prompt difficulty elevates entropy across all tokens in a sequence. Together, these results demonstrate that batch-level entropy statistics systematically prioritize tokens from hard prompts and early positions, regardless of whether those tokens represent genuine decision points.

Table 1: Prompt difficulty bias in token entropy on Llama-3.2-3B-Instruct.

	Global Entropy	Proximal Entropy
Hard	0.315	0.00987
Moderate	0.270	0.00987
Easy	0.221	0.00989

To understand why, consider the effect of prompt difficulty alone. Let $\bar{H}_i = \frac{1}{T_i}\sum_{t=1}^{T_i} H_{i,t}$ denote the mean entropy of rollout i, which serves as a measure of overall prompt difficulty. Each token's entropy decomposes as

$$H_{i,t} = \underbrace{\bar{H}_i}_{\text{sequence-level}} + \underbrace{(H_{i,t} - \bar{H}_i)}_{\text{token-specific}}, \tag{5}$$

where the token-specific component measures how surprising a token is relative to its own sequence, serving as a more reliable indicator of importance. Global methods, however, rank tokens by $H_{i,t}$ across the entire batch, allowing tokens to receive disproportionate credit simply because they belong to a high-entropy sequence. Accounting for the positional bias only compounds this problem, as the systematic decay of entropy over the course of generation means that early tokens receive additional unwarranted credit due to their position alone.

These observations motivate a token importance measure that accounts for both prompt difficulty and positional variation. Rather than asking *which tokens have high entropy relative to the global distribution*, we should ask *which tokens have high entropy relative to their own local context*. In the next section, we formalize this local reference frame and introduce PEPO, which replaces global entropy statistics with a sliding window centered around each token.

4 Proximal Entropy Policy Optimization

The key idea behind PEPO is to replace the global entropy statistics used by prior methods with a local measure we call *proximal entropy*. For each token, proximal entropy captures how uncertain the model is at that position relative to its neighboring tokens, rather than relative to the entire batch. This provides a per-token importance signal that is invariant to prompt difficulty and position.

Proximal entropy. Given the entropy sequence $(H_{i,1}, \ldots, H_{i,T_i})$ for rollout i, we define the proximal entropy of token t using a softmax over a sliding window of size W centered on t:

$$e_{i,t} = \frac{\exp(H_{i,t})}{\sum_{k \in \mathcal{W}(t)} \exp(H_{i,k})}, \tag{6}$$

where $\mathcal{W}(t) = \{k : |k - t| \leq W/2\}$ is the set of token positions within the window. The softmax formulation ensures that a token's proximal entropy reflects its uncertainty relative to its neighbors, not its absolute value. A token with moderately high entropy in an otherwise low-entropy region receives a large proximal entropy, while a token with the same entropy surrounded by equally uncertain neighbors does not.

Per-token advantage weighting. We use proximal entropy to modulate the GRPO advantage. Since each token's proximal entropy is computed over its own local window, the values do not sum to one across the sequence. We therefore normalize and scale them to preserve the total advantage magnitude:

$$\hat{A}_{i,t} = c_{i,t} \cdot A_i, \quad \text{where} \quad c_{i,t} = \frac{e_{i,t}}{\sum_{k=1}^{T_i} e_{i,k}} \cdot T_i. \tag{7}$$

This ensures that $\sum_t c_{i,t} = T_i$, so the total advantage magnitude is identical to that of standard GRPO. In practice, we apply the proximal entropy weighting only to rollouts with positive rewards, since amplifying updates on high-entropy tokens within failed rollouts would discourage exploration at those decision points rather than reinforce successful ones. The PEPO objective is then

$$\mathcal{J}_{\text{PEPO}}(\theta) = \frac{1}{\sum_{j=1}^{G} T_j} \sum_{i=1}^{G} \sum_{t=1}^{T_i} \min\big(\rho_{i,t}(\theta)\, \hat{A}_{i,t},\, \text{clip}(\rho_{i,t}(\theta), 1-\varepsilon_l, 1+\varepsilon_h)\, \hat{A}_{i,t}\big). \tag{8}$$

This is identical to the standard GRPO objective in Eq. (2), with the only difference being that the uniform advantage A_i is replaced by the entropy-weighted $\hat{A}_{i,t}$.

Theoretical justification. We now show that proximal entropy formally removes the confounders identified in Section 3.2.

Proposition 1. *The proximal entropy $e_{i,t}$ in Eq. (6) satisfies:*

Table 2: Main results on mathematical reasoning benchmarks. Trained methods are reported as mean ± sample standard deviation (SD) over three independent runs. We report avg@1 on MATH500 and avg@16 on AMC, AIME 2024, and AIME 2025; the Mean column summarizes each run's four benchmark scores. Best results per model are in **bold**.

Model	Method	AIME2024	AIME2025	AMC	MATH500	Mean
Qwen3-1.7B	Base	14.02 ±0.82	17.58 ±0.74	45.21 ±0.65	75.53 ±0.48	38.09 ±0.42
	GRPO	16.58 ±0.44	18.65 ±0.62	50.28 ±0.99	80.27 ±1.01	41.45 ±0.50
	Entropy Adv.	21.44 ±1.81	19.48 ±1.72	52.52 ±1.06	81.40 ±1.44	43.71 ±0.52
	80/20	20.97 ±1.48	20.83 ±0.75	52.13 ±0.44	80.73 ±1.50	43.67 ±0.30
	PEPO (Ours)	**21.74 ±1.06**	**22.43 ±0.67**	**55.26 ±1.11**	**82.67 ±0.31**	**45.53 ±0.46**
Qwen3-4B	Base	25.62 ±0.95	20.14 ±0.81	55.39 ±0.72	79.80 ±0.54	45.24 ±0.51
	GRPO	32.75 ±0.54	23.54 ±0.76	59.96 ±0.68	84.33 ±1.22	50.15 ±0.43
	Entropy Adv.	35.88 ±1.54	26.21 ±1.22	67.83 ±0.57	89.73 ±0.23	54.91 ±0.23
	80/20	35.35 ±2.47	30.62 ±0.21	67.25 ±1.53	89.73 ±1.15	55.74 ±0.86
	PEPO (Ours)	**39.24 ±1.27**	**31.81 ±0.13**	**69.58 ±0.59**	**91.40 ±1.83**	**58.01 ±0.14**
Llama-3.2-3B-Instruct	Base	5.69 ±0.45	0.62 ±0.18	21.49 ±0.52	45.93 ±0.61	18.43 ±0.28
	GRPO	7.22 ±0.84	0.90 ±0.52	21.43 ±0.57	47.67 ±1.33	19.31 ±0.20
	Entropy Adv.	8.26 ±0.84	0.56 ±0.24	23.64 ±0.74	47.60 ±0.60	20.02 ±0.44
	80/20	8.60 ±0.61	0.49 ±0.12	22.57 ±0.91	47.67 ±0.50	19.83 ±0.03
	PEPO (Ours)	**9.10 ±0.52**	**1.46 ±0.55**	**24.12 ±0.36**	**49.33 ±0.12**	**21.00 ±0.15**

(i) If all entropies in a sequence are shifted by a sequence-dependent constant δ_i, then $e_{i,t}$ is unchanged.

(ii) If entropies in a sequence follow a smooth monotonic trend, then $e_{i,t}$ is approximately invariant to the trend for sufficiently small W.

Property (i) implies that proximal entropy is invariant to prompt difficulty, as all tokens in a sequence share the same sequence-level component $\bar{H}_i$. Property (ii) implies robustness to positional entropy trends, which manifest as a smooth decay over the course of generation. Together, these properties ensure that proximal entropy isolates the token-specific signal from both confounders. The full proof is provided in Appendix B.

We note that Figure 2 and Table 1 empirically corroborate both properties. As shown in Figure 2, global entropy deviates by up to 80 points from its sequence-level mean across generation percentiles, while proximal entropy remains flat throughout, confirming the positional invariance of property (ii). Table 1 further shows that global entropy varies from 0.221 (easy) to 0.315 (hard) across prompt difficulty groups, whereas proximal entropy remains constant at ≈0.00988 across all three groups, confirming the shift invariance of property (i).

5 Experiments

5.1 Experimental Setup

Models. We evaluate on three base models spanning two model families: Qwen3-1.7B, Qwen3-4B [Yang et al., 2025], and Llama-3.2-3B-Instruct [Grattafiori et al., 2024].

Training. For the main experiments, we use the ROLL framework [Wang et al., 2025b] with the DeepMath-103K dataset [He et al., 2025]. For single-stream RL experiments (Section 5.4), we use the VeRL framework [Sheng et al., 2024] with the DAPO-Math-17K dataset [Yu et al., 2025]. All methods are implemented on top of GRPO [Shao et al., 2024] with $G = 8$ rollouts per prompt, a learning rate of 1×10^{-6}, clipping threshold $\varepsilon_h = 0.28$, and 500 training steps, and follow the default library configurations. For PEPO, we set the window size $W = 101$ across all experiments. The average response length during training ranges from approximately 2000 to 3000 tokens depending on the model, so the window covers roughly 3–5% of each sequence. Full hyperparameter details are provided in Appendix.

Table 3: Effect of substituting proximal entropy into 80/20. Each cell reports the mean ± sample SD over three independent runs. All other components of 80/20 remain unchanged. PEPO results from Table 2 are included for reference. Best results within each model are bolded.

Model	Method	AIME2024	AIME2025	AMC	MATH500	Mean
Qwen3-1.7B	80/20	20.97 ± 1.48	20.83 ± 0.75	52.13 ± 0.44	80.73 ± 1.50	43.67 ± 0.30
	80/20 + Proximal Entropy	**22.01 ± 0.87**	20.69 ± 0.24	53.36 ± 0.78	82.27 ± 0.31	44.58 ± 0.43
	PEPO (Ours)	21.74 ± 1.06	**22.43 ± 0.67**	**55.26 ± 1.11**	**82.67 ± 0.31**	**45.53 ± 0.46**
Qwen3-4B	80/20	35.35 ± 2.47	30.62 ± 0.21	67.25 ± 1.53	89.73 ± 1.15	55.74 ± 0.86
	80/20 + Proximal Entropy	36.18 ± 2.61	31.60 ± 0.48	69.38 ± 1.41	90.33 ± 0.31	56.87 ± 0.56
	PEPO (Ours)	**39.24 ± 1.27**	**31.81 ± 0.13**	**69.58 ± 0.59**	**91.40 ± 1.83**	**58.01 ± 0.14**

Baselines. We compare PEPO against three baselines: (1) standard GRPO [Shao et al., 2024] with uniform token weighting, (2) 80/20 [Wang et al., 2025a], which updates only the top 20% of tokens by global entropy, and (3) Entropy-augmented advantage [Cheng et al., 2025a], which augments the advantage function with a global entropy term.

Evaluation. We evaluate on MATH500 [Lightman et al., 2024], AMC [Mathematical Association of America, 2023], AIME 2024 [Mathematical Association of America, 2024], and AIME 2025 [Mathematical Association of America, 2025]. We report avg@1 accuracy on MATH500 and avg@16 accuracy on AMC, AIME 2024, and AIME 2025 over three runs. All detailed per-run results are provided in the Appendix.

5.2 Main Results

Table 2 presents the main results across all models and benchmarks. PEPO consistently outperforms both GRPO and batch-level entropy methods across all three models spanning two distinct model families, demonstrating that the improvements are robust and not an artifact of a particular architecture.

On Qwen3-4B, PEPO achieves the largest improvements, outperforming GRPO by 6.49%p on AIME 2024 and 9.62%p on AMC, and surpassing 80/20 by 2.27%p in mean accuracy across all benchmarks. On Qwen3-1.7B, PEPO improves over GRPO by 5.16%p on AIME 2024 and 4.98%p on AMC, with a mean accuracy gain of 1.82%p over Entropy Advantage. Notably, on Llama-3.2-3B-Instruct, 80/20 and Entropy Adv. degrade below GRPO on AIME 2025 (0.41%p and 0.34%p), while PEPO improves over GRPO on all benchmarks. The consistent gains of PEPO, even in cases where global entropy methods hurt performance, suggest that proximal entropy provides a more reliable token importance signal across model families.

5.3 Proximal Entropy as a General Formulation

To isolate the contribution of the proximal entropy formulation from the rest of PEPO's design, we substitute our window-relative entropy into 80/20, replacing its global entropy ranking with proximal entropy while keeping all other components unchanged. Concretely, instead of selecting the top 20% of tokens by global entropy across the batch, we select the top 20% by proximal entropy within the same batch. This tests whether proximal entropy is indeed a more accurate measure of token importance compared to global entropy. Table 3 shows the results.

Replacing global entropy with proximal entropy improves 80/20 across benchmarks on both models, with no other changes to the algorithm. This confirms that proximal entropy is a more accurate measure of token importance regardless of the algorithm it is embedded in. PEPO, which further replaces binary token selection with continuous entropy-based weighting, achieves the best overall performance across both models.

5.4 Transfer to Single-Stream RL

To test whether the proximal entropy formulation transfers beyond GRPO, we apply it to Single-stream Policy Optimization (SPO) [Xu and Ding, 2025], where each prompt

produces only a single rollout. We refer to this variant as PESPO (Proximal Entropy SPO). We use the same window size $W = 101$ and all other hyperparameters as in the main experiments, without any task-specific tuning. Table 4 shows the results. Both global entropy methods degrade below the SPO baseline in mean accuracy, with S-Entropy Adv. dropping to 54.65% and S-80/20 to 55.99%, compared to SPO's 57.13% in mean accuracy. The degradation is particularly pronounced on AIME 2025, where S-Entropy Adv. falls to 21.80%, nearly 10%p below SPO. This is consistent with the failure mode identified in Section 3.2. Without multiple rollouts per prompt, the batch contains a more diverse set of prompts, amplifying the prompt difficulty bias in global entropy statistics. PESPO, in contrast, achieves the highest mean accuracy (58.93%) and improves over SPO on AIME 2024 by 6.04%p, demonstrating that proximal entropy transfers to single-stream settings without hyperparameter adjustment. This provides further evidence that proximal entropy effectively captures token importance even when prompt diversity is large, whereas global entropy does not.

Table 4: Results on single-stream RL (SPO) with Qwen3-4B. Cells show the mean followed by sample SD in smaller type, computed over three independent runs. All methods are implemented on top of SPO. PESPO uses the same hyperparameters as in the main experiments without modification.

Method	AIME2024	AIME2025	AMC	MATH500	Mean
SPO	35.70 $\pm$ 1.97	31.39 $\pm$ 0.44	70.81 $\pm$ 1.28	90.60 $\pm$ 0.53	57.13 $\pm$ 0.42
S-Entropy Adv.	38.89 $\pm$ 1.26	21.80 $\pm$ 0.67	68.42 $\pm$ 0.43	89.47 $\pm$ 0.31	54.65 $\pm$ 0.58
S-80/20	35.28 $\pm$ 1.15	27.64 $\pm$ 1.05	70.06 $\pm$ 0.93	**91.00 $\pm$ 0.40**	55.99 $\pm$ 0.43
PESPO (Ours)	**41.74 $\pm$ 0.52**	**31.60 $\pm$ 0.79**	**71.71 $\pm$ 1.06**	90.67 $\pm$ 0.31	**58.93 $\pm$ 0.48**

5.5 Analysis

Token intervention. We test whether the entropy rankings identify consequential positions using forward cross-sampling, following prior token-replacement analyses [Meng et al., 2026, Huang et al., 2026]. On AIME 2024 avg@16, Qwen3-4B Base is the primary policy and a fixed GRPO checkpoint is the intervention policy. We compute raw entropy and proximal entropy rankings from the Base policy; at each selected position, we replace the Base token with a sample from GRPO and continue generation autoregressively. The intervention policy, replacement operator, evaluation set, and token budget are held fixed, so only the ranking statistic changes. Table 5 reports the resulting accuracy.

Table 5: Forward token-replacement cross-sampling on AIME 2024 (avg@16, Qwen3-4B). Entries are accuracy (%); budget is the fraction of positions intervened on.

Budget	Entropy	Proximal entropy
0%	25.62	25.62
5%	27.34	**31.62**
10%	29.91	**33.33**
15%	30.76	**33.33**
20%	32.47	**33.33**
30%	**33.33**	**33.33**

At a 5% replacement budget, proximal entropy reaches 31.62% accuracy versus 27.34% for entropy, a 4.28-point gap. At 10%, it reaches the GRPO checkpoint's 33.33% accuracy; entropy reaches that level only at 30%. These results provide evidence that proximal entropy is a more accurate measure of token importance than entropy in this setting, as it allows the GRPO checkpoint's full accuracy to be recovered with a smaller replacement budget, further supporting the claims in the main section.

Window size. We ablate the window size W on Qwen3-4B, comparing $W \in \{51, 101, 151\}$. Results are shown in Table 6. Performance is relatively robust across all three window sizes, suggesting that PEPO is not sensitive to this hyperparameter. This is consistent with the local smoothness assumption in Proposition 1. As long as the window is small enough relative to the sequence length, the positional trend is approximately cancelled. Among the three, $W = 101$ achieves the best results, which we use as the default across all experiments.

Table 6: Window-size ablation on Qwen3-4B. Entries report mean accuracy $\pm$ sample SD over three independent runs.

W	AIME2024	AIME2025	AMC	MATH500
$W = 51$	36.25 $\pm$ 1.50	30.62 $\pm$ 1.30	68.39 $\pm$ 0.59	90.67 $\pm$ 0.42
$W = 101$	39.24 $\pm$ 1.27	31.81 $\pm$ 0.13	69.58 $\pm$ 0.59	91.40 $\pm$ 1.83
$W = 151$	36.81 $\pm$ 0.24	28.27 $\pm$ 1.26	69.35 $\pm$ 0.49	92.27 $\pm$ 0.46

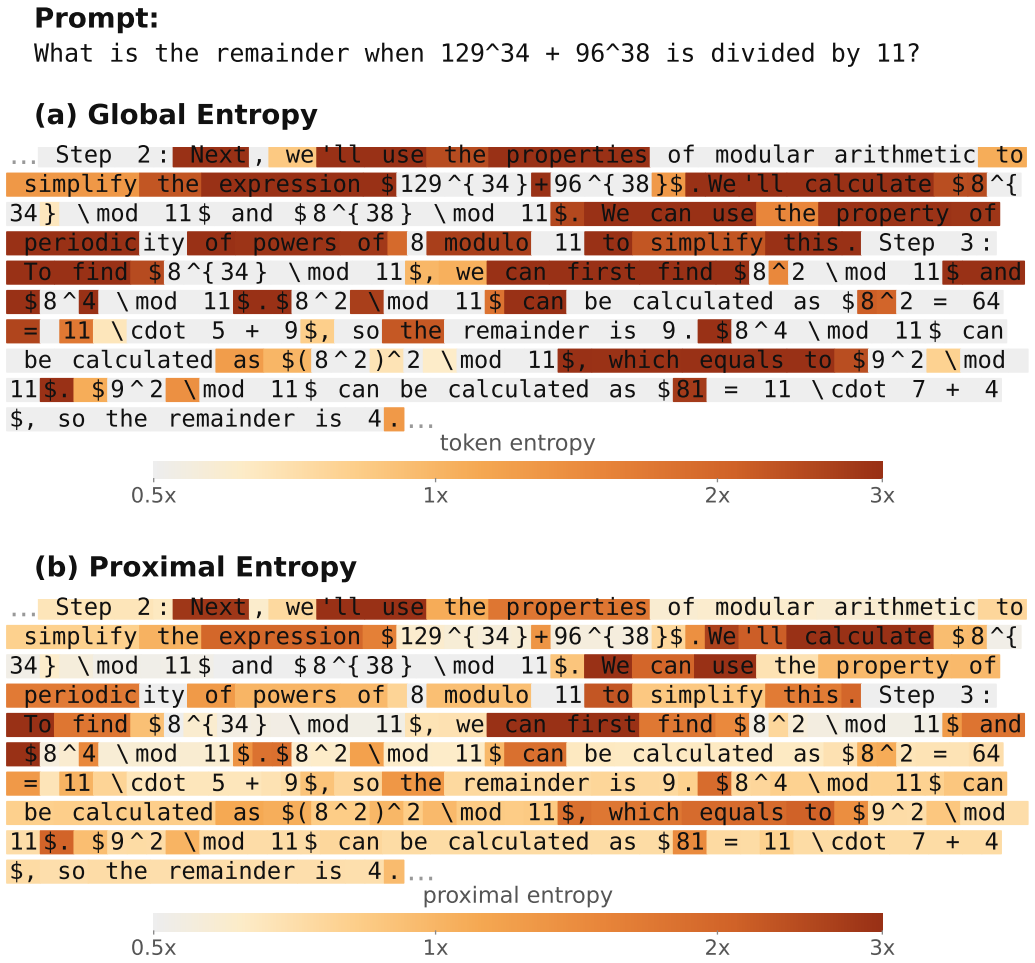


Figure 3: Qualitative comparison of token weighting on a Qwen3-4B rollout. **(a)** Global entropy assigns broad, near-uniform weight across structural and decision tokens alike (e.g., the entire phrases "We'll calculate" and "property of periodicity"). **(b)** Proximal entropy evaluates each token relative to its local neighborhood, producing fine-grained sub-phrase weighting that singles out the tokens that steer the reasoning trajectory (e.g., *calculate*, *periodicity*) while still preserving signal across the derivation steps.

Qualitative analysis. Figure 3 (Appendix) visualizes the token-level scores produced by global entropy and proximal entropy on a Qwen3-4B rollout for the modular arithmetic problem $129^{34} + 96^{38} \bmod 11$, where each token is colored by its score relative to the mean of each metric. Two properties of proximal entropy stand out. First, proximal entropy produces *fine-grained, sub-phrase weighting* that global entropy cannot. In "We'll calculate," for instance, global entropy lights up nearly uniformly across all three tokens, whereas proximal entropy distinguishes between them. *We* and *calculate* receive substantially more weight than *'ll*, with *calculate* receiving the strongest weight as the token that commits to the action. The same effect appears in "property of periodicity," where global entropy assigns elevated weight to all three content words, while proximal entropy concentrates on *periodicity*, the noun that names the mathematical principle being invoked. This local reference frame allows proximal entropy to identify the specific tokens that steer the reasoning trajectory, even within short phrases where every token shares similar local context.

Second, proximal entropy does not ignore the derivation itself. The execution tokens that carry out the modular arithmetic — "$8^2 = 64 = 11 \cdot 5 + 9$," "$81 = 11 \cdot 7 + 4$" — receive visibly nontrivial weight under proximal entropy, even though their absolute entropy is near zero (each digit follows deterministically from the formula structure). Global entropy, in contrast, washes these tokens out

entirely. Proximal entropy's window-relative formulation preserves signal across the derivation while still concentrating attention on the higher-leverage decision points.

Additional evaluation and ablations, including held-out code generation, the weighting design, and temperature sensitivity, are reported in Appendix F.

6 Conclusion

We introduced Proximal Entropy Policy Optimization (PEPO), which replaces global entropy statistics with *proximal entropy*, a local measure of token importance for value-model-free RLVR. By computing entropy relative to a sliding window around each token, PEPO identifies tokens that are genuinely important given their local context, rather than tokens that merely reside in high-entropy regions of the batch. We formally showed that this formulation is invariant to prompt difficulty and approximately invariant to positional entropy trends (Proposition 1), directly addressing the confounders we identified in global entropy methods. Experiments across Qwen3-1.7B, Qwen3-4B, and Llama-3.2-3B-Instruct demonstrated consistent improvements over GRPO and batch-level entropy methods on mathematical reasoning benchmarks. On a held-out LiveCodeBench split, Qwen3-4B with PEPO also improved pass@1 over GRPO and 80/20. We further showed that the proximal entropy formulation generalizes beyond PEPO itself. Substituting it into 80/20 yields gains without any other changes, and applying it to single-stream RL (PESPO) improves performance where global entropy degrades.

Limitations and future work. Our main benchmark suite focuses on mathematical reasoning. The additional coding evaluation covers one held-out LiveCodeBench split and Qwen3-4B, so broader evaluation across coding benchmarks and model families is needed to establish how well PEPO generalizes to code generation, agentic tool use, and multi-turn dialogue. We did not compare against value-based methods such as PPO, which provide a stronger but more expensive form of credit assignment. Combining proximal entropy with a learned value model could offer complementary benefits. More broadly, the local reference frame could be applied beyond advantage weighting, for instance, to guide exploration strategies or to design entropy-aware curricula that adapt at the token level rather than the prompt level.

Acknowledgments and Disclosure of Funding

This work was supported by the Korean Government through grants from IITP (RS-2021-II211343 and RS-2025-25442338) and NRF (RS-2026-25500579).

Competing interests: The authors declare no competing interests.

References

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, et al. MiniMax-M1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025. Introduces CISPO.

Shuang Chen, Yue Guo, Yimeng Ye, Shijue Huang, Wenbo Hu, Haoxi Li, Manyuan Zhang, Jiayu Chen, Song Guo, and Nanyun Peng. ARES: Multimodal adaptive reasoning via difficulty-aware token-level entropy shaping. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

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. *arXiv preprint arXiv:2506.14758*, 2025a.

Jie Cheng, Gang Xiong, Ruixi Qiao, Lijun Li, Chao Guo, Junle Wang, Yisheng Lv, and Fei-Yue Wang. Stop summation: Min-form credit assignment is all process reward model needs for reasoning. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025b.

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

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*, 2025a.

Xu Guo. Measuring reasoning utility in LLMs via conditional entropy reduction. *arXiv preprint arXiv:2508.20395*, 2025.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in RL for large language models. *arXiv preprint arXiv:2505.23564*, 2025b.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. DeepMath-103K: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025.

Kexin Huang, Haoming Meng, Junkang Wu, Jinda Lu, Chiyu Ma, Ziqian Chen, Xue Wang, Bolin Ding, Jiancan Wu, Xiang Wang, Xiangnan He, Guoyin Wang, and Jingren Zhou. On the direction of RLVR updates for LLM reasoning: Identification and exploitation. *arXiv preprint arXiv:2603.22117*, 2026.

Zhengbo Jiao, Shaobo Wang, Zifan Zhang, Wei Wang, Bing Zhao, Hu Wei, and Linfeng Zhang. Credit where it is due: Cross-modality connectivity drives precise reinforcement learning for MLLM reasoning. *arXiv preprint arXiv:2602.11455*, 2026.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, 2023.

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

Yang Li, Zhichen Dong, Yuhan Sun, Weixun Wang, Shaopan Xiong, Yijia Luo, Jiashun Liu, Han Lu, Jiamang Wang, Wenbo Su, Bo Zheng, and Junchi Yan. Attention illuminates LLM reasoning: The preplan-and-anchor rhythm enables fine-grained policy optimization. *arXiv preprint arXiv:2510.13554*, 2025.

Yugu Li, Zehong Cao, Jianglin Qiao, and Siyi Hu. SSVPO: Effective step-level credit assignment for RL training of language models. In *ICLR 2026 Conference (under review)*, 2026. OpenReview id g33DGvnHYd.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Mathematical Association of America. AMC 2023: American mathematics competitions. `https://artofproblemsolving.com/wiki/index.php/AMC_Problems_and_Solutions`, 2023.

Mathematical Association of America. AIME 2024: American invitational mathematics examination. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`, 2024.

Mathematical Association of America. AIME 2025: American invitational mathematics examination. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`, 2025.

Haoming Meng, Kexin Huang, Shaohang Wei, Chiyu Ma, Shuo Yang, Xue Wang, Guoyin Wang, Bolin Ding, and Jingren Zhou. Sparse but critical: A token-level analysis of distributional shifts in RLVR fine-tuning of LLMs. In *International Conference on Learning Representations*, 2026.

Shuaiyi Nie, Siyu Ding, Wenyuan Zhang, Linhao Yu, Tianmeng Yang, Yao Chen, Weichong Yin, Yu Sun, Hua Wu, and Tingwen Liu. ATTNPO: Attention-guided process supervision for efficient reasoning. *arXiv preprint arXiv:2602.09953*, 2026.

Prasanna Parthasarathi, Mathieu Reymond, Boxing Chen, Yufei Cui, and Sarath Chandar. GRPO-λ: Credit assignment improves LLM reasoning. *arXiv preprint arXiv:2510.00194*, 2025.

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

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. *arXiv preprint arXiv:2409.19256*, 2024.

Hongze Tan, Zihan Wang, Jianfei Pan, Jinghao Lin, Hao Wang, Yifan Wu, Tao Chen, Zhihang Zheng, Zhihao Tang, and Haihua Yang. GTPO and GRPO-S: Token and sequence-level reward shaping with policy entropy. *arXiv preprint arXiv:2508.04349*, 2025.

Hieu Tran, Zonghai Yao, and Hong Yu. Exploiting tree structure for credit assignment in RL training of LLMs. *arXiv preprint arXiv:2509.18314*, 2025.

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. *arXiv preprint arXiv:2506.01939*, 2025a.

Tao Wang, Suhang Zheng, and Xiaoxiao Xu. RTMC: Step-level credit assignment via rollout trees. *arXiv preprint arXiv:2604.11037*, 2026.

Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, et al. ROLL: Reinforcement learning optimization for large-scale learning – an efficient and user-friendly scaling library. *arXiv preprint arXiv:2506.06122*, 2025b.

Can Xie, Ruotong Pan, Xiangyu Wu, Yunfei Zhang, Jiayi Fu, Tingting Gao, and Guorui Zhou. Unlocking exploration in RLVR: Uncertainty-aware advantage shaping for deeper reasoning. *arXiv preprint arXiv:2510.10649*, 2025a.

Guofu Xie, Yunsheng Shi, Hongtao Tian, Ting Yao, and Xiao Zhang. CAPO: Credit assignment policy optimization. *arXiv preprint arXiv:2508.02298*, 2025b.

Zhongwen Xu and Zihan Ding. Single-stream policy optimization. *arXiv preprint arXiv:2509.13232*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Song Yu, Li Li, Wenwen Zhao, and Zhisheng Yang. ERPO: Token-level entropy-regulated policy optimization for large reasoning models. *arXiv preprint arXiv:2603.28204*, 2026.

Xingjian Zhang, Siwei Wen, Wenjun Wu, and Lei Huang. EDGE-GRPO: Entropy-driven GRPO with guided error correction for advantage diversity. *arXiv preprint arXiv:2507.21848*, 2025.

Zheng Zhang, Ao Lu, Yuanhao Zeng, Ziwei Shan, Jinjin Guo, Lufei Li, Yexin Li, and Kan Ren. Grad2Reward: From sparse judgment to dense rewards for improving open-ended LLM reasoning. *arXiv preprint arXiv:2602.01791*, 2026.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, Fang Wan, and Furu Wei. Geometric-mean policy optimization. *arXiv preprint arXiv:2507.20673*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Jiaru Zou, Ling Yang, Jingwen Gu, Jiahao Qiu, Ke Shen, Jingrui He, and Mengdi Wang. ReasonFlux-PRM: Trajectory-aware PRMs for long chain-of-thought reasoning in LLMs. *arXiv preprint arXiv:2506.18896*, 2025.

Appendix

A Broader Impacts

PEPO is a methodological contribution to value-model-free RLVR that improves token-level credit assignment without additional rollouts, value models, or process reward models. As a general-purpose post-training technique, its societal impact is largely inherited from the broader trajectory of LLM reasoning research rather than tied to a specific deployment.

On the positive side, PEPO reduces the compute required to obtain a given level of reasoning performance: it matches or exceeds batch-level entropy methods at the same training budget (Table 9) and transfers to single-stream RL, where it removes the need for G parallel rollouts per prompt. Lower training cost lowers the barrier to entry for academic and resource-constrained groups working on reasoning models, and reduces the energy footprint of RL post-training.

The negative considerations are those common to any technique that improves LLM reasoning. Stronger reasoning capabilities accelerate both beneficial applications (scientific assistance, education, code generation) and potentially harmful ones (more persuasive misinformation, automated vulnerability discovery, misuse in high-stakes decisions). PEPO does not introduce a new capability axis or unlock qualitatively new behaviors; it makes existing RLVR pipelines more sample- and credit-efficient. We do not release any new pretrained models or scraped datasets, and all experiments build on publicly available base models and mathematical reasoning datasets that have been released by their original authors with their own usage terms. We see no direct path from this work to specific harmful applications beyond those already discussed in the broader literature on LLM post-training.

B Proof of Proposition 1

We restate the proposition for convenience.

Proposition 2. *The proximal entropy $e_{i,t}$ (Eq. 6) satisfies:*

(i) If all entropies in a sequence are shifted by a sequence-dependent constant δ_i, then $e_{i,t}$ is unchanged.

(ii) If entropies in a sequence follow a smooth monotonic trend, then $e_{i,t}$ is approximately invariant to the trend for sufficiently small W.

B.1 Proof of (i): Invariance to Constant Shift

Let $\tilde{H}_{i,t} = H_{i,t} + \delta_i$ for all t and some sequence-dependent constant $\delta_i \in \mathbb{R}$. The corresponding proximal entropy is

$$\begin{aligned}\tilde{e}_{i,t} &= \frac{\exp(\tilde{H}_{i,t})}{\sum_{k\in\mathcal{W}(t)}\exp(\tilde{H}_{i,k})} = \frac{\exp(H_{i,t}+\delta_i)}{\sum_{k\in\mathcal{W}(t)}\exp(H_{i,k}+\delta_i)} \\ &= \frac{\exp(\delta_i)\exp(H_{i,t})}{\exp(\delta_i)\sum_{k\in\mathcal{W}(t)}\exp(H_{i,k})} = \frac{\exp(H_{i,t})}{\sum_{k\in\mathcal{W}(t)}\exp(H_{i,k})} = e_{i,t}. \end{aligned} \tag{9}$$

Since all tokens within a window $\mathcal{W}(t)$ belong to the same rollout i, they share the same δ_i, which factors out and cancels. This holds for any sequence-dependent shift, so proximal entropy is invariant to differences in overall entropy level across sequences.

B.2 Proof of (ii): Approximate Invariance to Positional Trends

We decompose each token's entropy as

$$H_{i,t} = f(t) + r_{i,t}, \tag{10}$$

where $f\colon [T_i] \to \mathbb{R}$ is a smooth, monotonic, deterministic positional trend shared across rollouts and $r_{i,t}$ is the residual that captures token-specific variation. Our goal is to show that $e_{i,t}$ is approximately independent of f under mild regularity conditions.

Setup. Fix a token position t and consider the window $\mathcal{W}(t) = \{k : |k - t| \leq W/2\}$. Since f is monotonic, $f'(t)$ does not change sign within the window, so the linear term dominates the local variation of f and a first-order Taylor expansion of f around t gives

$$f(k) = f(t) + f'(t)(k - t) + \mathcal{O}\left(\frac{W^2}{T_i^2}\right), \tag{11}$$

where we have used $|k - t| \leq W/2$ and the fact that f varies over the full sequence length T_i, so $\sup |f''| = \mathcal{O}(1/T_i^2)$. Monotonicity ensures that the residual term captures only curvature rather than direction-changing oscillations, so the approximation is accurate when $W \ll T_i$, i.e., the positional trend is approximately linear within each window.

Substitution. Substituting Eq. (11) into the definition of $e_{i,t}$ and dropping the higher-order terms:

$$\begin{aligned} e_{i,t} &= \frac{\exp(H_{i,t})}{\sum_{k\in\mathcal{W}(t)} \exp(H_{i,k})} = \frac{\exp\big(f(t) + r_{i,t}\big)}{\sum_{k\in\mathcal{W}(t)} \exp\big(f(k) + r_{i,k}\big)} \\ &\approx \frac{\exp\big(f(t) + r_{i,t}\big)}{\sum_{k\in\mathcal{W}(t)} \exp\big(f(t) + f'(t)(k - t) + r_{i,k}\big)} \\ &= \frac{\exp(r_{i,t})}{\sum_{k\in\mathcal{W}(t)} \exp\big(f'(t)(k - t) + r_{i,k}\big)}, \end{aligned} \tag{12}$$

where the last step uses the translation-invariance property established in (i) to cancel $f(t)$.

Small-gradient regime. Eq. (12) still contains the linear term $f'(t)(k - t)$. We show that this term is negligible when the trend slope is small relative to the inverse window size. For each $k \in \mathcal{W}(t)$, we have $|f'(t)(k - t)| \leq |f'(t)| \cdot W/2$. When $|f'(t)| \cdot W/2 \ll 1$, a first-order expansion of the exponential gives

$$\exp\big(f'(t)(k - t) + r_{i,k}\big) \approx \exp(r_{i,k})\big(1 + f'(t)(k - t)\big). \tag{13}$$

Substituting into the denominator of Eq. (12):

$$\sum_{k\in\mathcal{W}(t)} \exp\big(f'(t)(k - t) + r_{i,k}\big) \approx \sum_{k\in\mathcal{W}(t)} \exp(r_{i,k}) + f'(t) \sum_{k\in\mathcal{W}(t)} (k - t) \exp(r_{i,k}). \tag{14}$$

The second term is a weighted sum of offsets $(k - t)$, which range symmetrically from $-W/2$ to $+W/2$. Even without any assumption on the residuals, this term is $\mathcal{O}(f'(t) \cdot W)$ times the first term, and therefore negligible when $|f'(t)| \cdot W/2 \ll 1$. The same expansion applies to the numerator at $k = t$, where $(k - t) = 0$ and the linear correction vanishes exactly. We therefore obtain

$$e_{i,t} \approx \frac{\exp(r_{i,t})}{\sum_{k\in\mathcal{W}(t)} \exp(r_{i,k})}, \tag{15}$$

which depends only on the residuals and is independent of the trend f.

Summary. The conditions required are that the positional trend f is smooth and monotonic, and that $|f'(t)| \cdot W/2 \ll 1$, i.e., the trend is slowly varying within each window. Under these conditions, the proximal entropy $e_{i,t}$ is approximately invariant to f up to corrections of order $\mathcal{O}(f'(t) \cdot W)$ and the Taylor remainder from Eq. (11). Combined with property (i), this implies that $e_{i,t}$ isolates the token-specific signal from both prompt difficulty and positional entropy trends. □

Remark. The condition $|f'(t)| \cdot W/2 \ll 1$ is mild for typical reasoning rollouts. From Fig. 2, entropy decays from roughly 0.314 to 0.185 over the full sequence, giving $|f'| \approx 0.13/T_i$. For $T_i = 2000$ and $W = 101$, we have $|f'| \cdot W/2 \approx 0.004$, well within the small-gradient regime. The window size ablation in Fig 4 provides further empirical support, where performance is stable across $W \in \{31, 51, 101\}$, suggesting that the approximation holds well for the window sizes used.

C Implementation Details

C.1 Hyperparameters

We follow the default configurations of the ROLL framework [Wang et al., 2025b] for our main experiments and the VeRL framework [Sheng et al., 2024] for single-stream experiments. Table 7 reports the hyperparameters used for the main and drop-in experiments (Tables 2, 3, 6), and Table 8 reports those used for the single-stream RL experiments (Table 4).

Table 7: Hyperparameters for main experiments on ROLL framework. These settings are used for GRPO, 80/20, Entropy Adv., and PEPO across all three models.

Parameter	**Value**
Framework	ROLL
Training data	DeepMath-103K
Rollouts per prompt (G)	8
Learning rate	1×10^{-6}
Optimizer	AdamW
Lower clipping threshold (ε_l)	0.2
Upper clipping threshold (ε_h)	0.28
Training steps	500
Sampling temperature	0.6
Top-p	0.6
Maximum response length	4096 tokens
Window size W (PEPO)	101

Table 8: Hyperparameters for single-stream experiments on VeRL framework. These settings are used for SPO, S-80/20, S-Entropy Adv., and PESPO.

Parameter	**Value**
Framework	VeRL
Training data	DAPO-Math-17K
Rollouts per prompt	1 (single-stream)
Learning rate	1×10^{-6}
Optimizer	AdamW
Lower clipping threshold (ε_l)	0.2
Upper clipping threshold (ε_h)	0.28
Training steps	500
Sampling temperature	0.6
Top-p	0.6
Maximum response length	4096 tokens
Window size W (PESPO)	101

C.2 Window Edge Handling

The window $\mathcal{W}(t) = \{k : |k - t| \leq W/2\}$ extends beyond the sequence boundaries when $t < W/2$ or $t > T_i - W/2$. We use reflect padding to fill the missing positions, mirroring entropy values across the boundary. We chose reflect padding over zero padding because the latter would inject artificially low entropy values into the softmax denominator at boundary windows, distorting the relative weighting. Reflect padding instead uses values drawn from the same sequence, preserving the local entropy scale. This is also intuitively aligned with the proximal entropy formulation itself, which measures each token's entropy relative to its neighbors; reflecting the sequence at the boundary effectively treats nearby tokens on the opposite side of t as the missing neighbors, consistent with the local-context interpretation of $e_{i,t}$.

Boundary effects are confined to roughly $W/T_i \approx 5\%$ of tokens for our default $W = 101$ and typical sequence lengths, and we found this choice to work well empirically.

C.3 Software and Compute

All experiments are run on 2 NVIDIA H200 GPUs. We use the ROLL framework for the main experiments and the VeRL framework for single-stream experiments, with vLLM for rollout generation.

D Runtime Analysis

To verify that proximal entropy weighting adds negligible overhead, we measure the total wall-clock training time of PEPO and the baselines under identical hardware (2×H200), framework (ROLL), and training configuration (500 steps, $G = 8$, max length 4096). Table 9 reports the results for the Qwen3-1.7B and Qwen3-4B.

Table 9: Total wall-clock training time on 2×H200 GPUs. All methods are run with identical training configurations for 500 steps. PEPO adds negligible overhead over GRPO across both models.

Method	**Qwen3-1.7B**	**Qwen3-4B**
GRPO	1d 7h 32m	1d 19h 25m
80/20	1d 7h 28m	1d 20h 14m
Entropy Adv.	1d 8h 28m	1d 21h 3m
PEPO (Ours)	1d 7h 54m	1d 19h 45m

As shown in Table 9, all entropy-based methods (80/20, Entropy Adv., PEPO) run within minutes of GRPO across both models, well within the margin of variation expected from differences in random seeds and hardware scheduling.

E Additional Bias Analyses of Proximal Entropy

In Section 2.2 of the main paper, we showed that proximal entropy is empirically invariant to prompt difficulty (Table 1) and positional trends (Figure 2). In this section, we provide additional analyses showing that this invariance is robust across window sizes and is not an artifact of the softmax operation.

Invariance across window sizes and aggregation choices. Proposition 1(ii) establishes that proximal entropy is approximately invariant to positional trends for sufficiently small W. To empirically verify this, we plot the deviation of two window-based statistics from the sequence-level mean across generation percentiles for $W \in \{31, 51, 101\}$: (i) proximal entropy as defined in Eq. 6 and (ii) a simple windowed average of token entropies without the softmax operation. As shown in Figure 4, both statistics yield flat curves across all three window sizes, in contrast to global entropy (gray dashed) which decays substantially over the course of generation.

This result has two implications. First, the positional invariance of proximal entropy is robust to the choice of W, holding across the range of window sizes considered in our experiments. Second, the mitigation of positional bias is a property of the local windowing itself rather than the softmax operation: even a simple windowed average suffices to remove the positional trend. The softmax in PEPO serves a different purpose, which is to convert a window of entropies into a continuous, normalized per-token weight that emphasizes tokens with relatively higher entropy than their local neighbors. The two design choices (windowing and softmax-based weighting) are therefore separable, and the bias-removal property attaches to the former.

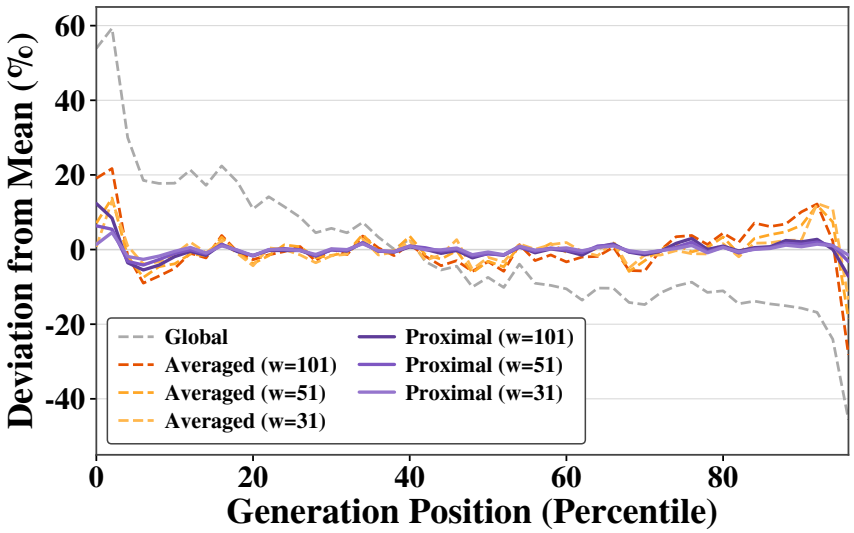


Figure 4: Deviation of windowed entropy statistics from their sequence-level mean across generation percentiles, for window sizes $W \in \{31, 51, 101\}$. We compare two aggregation choices: proximal entropy (solid purple), defined as a softmax over the window, and a simple windowed average (dashed orange). Both yield flat curves across all three window sizes, in contrast to global entropy (gray dashed) which decays substantially. This shows that the positional invariance is robust to the window size and is a property of the windowing itself rather than of the softmax operation. Computed over 992 rollouts on the validation set with Llama-3.2-3B-Instruct.

F Additional Evaluation and Ablations

F.1 Held-Out Code Generation: LiveCodeBench

To test transfer beyond mathematical reasoning, we evaluate the Qwen3-4B GRPO, 80/20, and PEPO checkpoints on the held-out `release_v6/code_generation_lite` split of LiveCodeBench, which contains 1,055 problems. LiveCodeBench was not included in RL training; the ROLL training mixture did include separate verifiable coding problems from KodCode. The three checkpoints share the same base model, training data, and training budget. We use the official LiveCodeBench evaluation implementation (`https://github.com/LiveCodeBench/LiveCodeBench`) with temperature 0.2, top-p 0.95, a maximum of 4,096 generated tokens, thinking mode enabled, and bfloat16 inference. Table 10 reports overall pass@1, and Table 11 reports the difficulty and platform breakdowns.

PEPO reaches 46.35% pass@1, a 14.60 percentage-point improvement over GRPO and 5.31 points over 80/20. Its gains over GRPO are positive across all three difficulty groups and all listed platforms. The Codeforces subset contains only nine problems, so that row should be interpreted cautiously.

Table 10: Held-out LiveCodeBench `release_v6/code_generation_lite` pass@1 point estimates for Qwen3-4B.

Method	Correct	Pass@1	Gain vs. GRPO
GRPO	335 / 1,055	31.75%	—
80/20	433 / 1,055	41.04%	+9.29 pp
PEPO	489 / 1,055	**46.35%**	**+14.60 pp**

Table 11: LiveCodeBench pass@1 breakdown by difficulty and problem platform. The Codeforces subset contains only 9 problems.

Subset	Problems	GRPO	80/20	PEPO	PEPO − GRPO
By difficulty					
Easy	322	76.09%	84.78%	**90.99%**	+14.91 pp
Medium	383	22.72%	37.60%	**43.86%**	+21.15 pp
Hard	350	0.86%	4.57%	**8.00%**	+7.14 pp
By platform					
AtCoder	602	31.23%	38.87%	**42.86%**	+11.63 pp
LeetCode	444	32.66%	43.92%	**50.68%**	+18.02 pp
Codeforces	9	22.22%	44.44%	**66.67%**	+44.44 pp

F.2 Weighting Design Ablation

We decompose the Qwen3-4B mean accuracy gain by changing one component at a time: global to proximal entropy for binary selection, binary selection to continuous weighting over all trajectories, and then positive-only application. Table 12 reports the aggregate results.

Table 12: Sequential weighting ablation on Qwen3-4B. Mean accuracy averages AIME 2024, AIME 2025, AMC, and MATH500; the first gain averages the benchmark-level differences in Table 3.

Configuration	Mean	Gain (pp)
Global entropy, binary selection (80/20)	55.74	—
Proximal entropy, binary selection	56.87	+1.14
Proximal entropy, continuous weighting on all rollouts	57.53	+0.66
Proximal entropy, continuous weighting on positive rollouts	**58.01**	+0.48

Localization adds 1.14 points, moving from binary to continuous weighting over all trajectories adds 0.66 points, and restricting continuous weighting to positive-reward rollouts adds a further 0.48 points. The first gain is the average of the four benchmark-level changes shown in Table 3; the all-rollout continuous-weighting result was reported without run-level uncertainty, so its comparison with positive-only weighting is descriptive. Even with all-rollout weighting, the mean remains 7.38 points above GRPO (57.53 versus 50.15).

F.3 Temperature Sensitivity

We vary the proximal-entropy softmax temperature τ on Llama-3.2-3B-Instruct while keeping the remaining settings fixed. Table 13 gives the reported point estimates.

Table 13: Temperature sensitivity for Llama-3.2-3B-Instruct. Results are the reported point estimates; no seed-level uncertainty was provided for this sweep. Best result per column is bolded.

τ	AIME24	AIME25	AMC	MATH500	Mean
0.5	8.75	1.25	23.72	**49.40**	20.78
1.0	**9.10**	**1.46**	**24.12**	49.33	**21.00**
2.0	8.54	**1.46**	23.72	49.00	20.68

Mean accuracy spans only 20.68–21.00 across the sweep, with the default $\tau = 1$ giving the highest reported mean.

Per-Run Experimental Results

Main Results (Table 2)

Table 14: Per-run accuracy for the main results in Table 2. Each row reports the avg@1 accuracy on MATH500 and avg@16 accuracy on AMC, AIME 2024, and AIME 2025 for a single run. Mean rows match the values reported in Table 2.

Model	Method	Run	AIME2024	AIME2025	AMC	MATH500
Qwen3-1.7B	GRPO	1	16.42	18.26	51.22	79.20
		2	17.07	19.36	50.38	81.20
		3	16.24	18.33	49.25	80.40
		Mean	16.58	18.65	50.28	80.27
	Entropy Adv.	1	22.25	20.53	51.61	79.80
		2	19.37	17.50	53.69	82.60
		3	22.71	20.41	52.26	81.80
		Mean	21.44	19.48	52.52	81.40
	80/20	1	22.29	21.46	51.95	79.20
		2	21.25	21.04	52.63	80.80
		3	19.37	20.00	51.80	82.20
		Mean	20.97	20.83	52.13	80.73
	PEPO	1	20.84	22.71	54.54	82.40
		2	22.91	21.67	56.54	83.00
		3	21.46	22.91	54.69	82.60
		Mean	21.74	22.43	55.26	82.67
Qwen3-4B	GRPO	1	32.68	22.92	60.25	83.00
		2	33.33	24.38	59.18	85.40
		3	32.25	23.31	60.45	84.60
		Mean	32.75	23.54	59.96	84.33
	Entropy Adv.	1	36.41	24.88	68.25	89.60
		2	37.08	26.46	67.18	90.00
		3	34.15	27.29	68.06	89.60
		Mean	35.88	26.21	67.83	89.73
	80/20	1	32.50	30.83	67.77	88.40
		2	36.87	30.62	68.46	90.40
		3	36.67	30.42	65.53	90.40
		Mean	35.35	30.62	67.25	89.73
	PEPO	1	38.13	31.88	70.17	91.80
		2	40.63	31.88	69.58	89.40
		3	38.96	31.66	68.99	93.00
		Mean	39.24	31.81	69.58	91.40
Llama-3.2-3B-Instruct	GRPO	1	6.46	0.83	21.69	48.00
		2	7.08	1.46	20.78	48.80
		3	8.12	0.42	21.83	46.20
		Mean	7.22	0.90	21.43	47.67
	Entropy Adv.	1	8.12	0.42	24.24	47.60
		2	9.17	0.42	23.86	48.20
		3	7.50	0.83	22.81	47.00
		Mean	8.26	0.56	23.64	47.60
	80/20	1	7.92	0.42	23.26	47.60
		2	9.12	0.62	21.53	48.20
		3	8.75	0.42	22.91	47.20
		Mean	8.60	0.49	22.57	47.67
	PEPO	1	8.54	1.67	23.71	49.40
		2	9.58	0.83	24.40	49.40
		3	9.17	1.87	24.24	49.20
		Mean	9.10	1.46	24.12	49.33

Proximal Entropy as a General Formulation (Table 3)

Table 15: Per-run accuracy for the drop-in proximal entropy results in Table 3. Mean rows match the values reported in Table 3.

Model	Method	Run	AIME2024	AIME2025	AMC	MATH500
Qwen3-1.7B	80/20 + Prox.	1	22.71	20.83	53.16	82.60
		2	21.04	20.42	52.71	82.20
		3	22.29	20.83	54.22	82.00
		Mean	22.01	20.69	53.36	82.27
Qwen3-4B	80/20 + Prox.	1	38.75	31.04	69.65	90.60
		2	33.54	31.88	70.63	90.40
		3	36.25	31.88	67.85	90.00
		Mean	36.18	31.60	69.38	90.33

Single-Stream RL (Table 4)

Table 16: Per-run accuracy for the single-stream RL results in Table 4 on Qwen3-4B. Mean rows match the values reported in Table 4.

Method	Run	AIME2024	AIME2025	AMC	MATH500
SPO	1	37.92	31.25	70.71	90.00
	2	35.00	31.04	72.14	90.80
	3	34.17	31.88	69.58	91.00
	Mean	35.70	31.39	70.81	90.60
S-80/20	1	36.46	26.67	68.98	90.60
	2	35.21	28.75	70.56	91.40
	3	34.17	27.50	70.63	91.00
	Mean	35.28	27.64	70.06	91.00
S-Entropy Adv.	1	38.75	22.29	68.30	89.80
	2	40.21	22.08	68.90	89.40
	3	37.71	21.04	68.07	89.20
	Mean	38.89	21.80	68.42	89.47
PESPO	1	41.25	31.04	71.76	90.60
	2	42.29	32.50	72.74	90.40
	3	41.67	31.25	70.63	91.00
	Mean	41.74	31.60	71.71	90.67

Window Size Ablation (Table 6)

Table 17: Per-run accuracy for the window size ablation in Table 6 on Qwen3-4B. Mean rows match the values reported in Table 6.

Window Size	Run	AIME2024	AIME2025	AMC	MATH500
$W = 51$	1	34.59	30.20	68.36	90.20
	2	37.50	29.59	67.82	91.00
	3	36.67	32.08	68.99	90.80
	Mean	36.25	30.62	68.39	90.67
$W = 151$	1	37.08	27.08	69.87	92.00
	2	36.67	29.59	68.89	92.00
	3	36.67	28.13	69.29	92.80
	Mean	36.81	28.27	69.35	92.27

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 (Section 1) state three contributions: (i) identifying the prompt-difficulty and positional biases of batch-level entropy, (ii) proposing PEPO with proximal entropy and proving its invariance to both confounders, and (iii) demonstrating consistent gains across three models on four benchmarks and showing that the proximal entropy formulation generalizes to other algorithms (80/20) and settings (single-stream RL). These claims are supported by Section 3.2 (bias analysis), Section 4 with Proposition 1 (theory), and Section 5 (experiments).

2. **Limitations**

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

   Answer: [Yes]

   Justification: Section 6 (Conclusion) explains that the main benchmark suite focuses on mathematical reasoning, while the additional coding evaluation covers one held-out LiveCodeBench split and Qwen3-4B. It also notes the lack of comparison against value-based methods such as PPO and identifies broader coding, agentic tool-use, and multi-turn dialogue evaluations as future work.

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: [Yes]

   Justification: Proposition 1 in Section 4 states the two invariance properties of proximal entropy. The full proof is provided in Appendix A, including the assumptions (smoothness and monotonicity of the positional trend, and the small-gradient regime $|f'(t)| \cdot W/2 \ll 1$). The proof verifies both the constant-shift invariance (A.1) and the approximate trend invariance (A.2) with explicit Taylor expansions and remainder bounds, and Appendix A includes a remark numerically verifying the small-gradient condition holds for typical reasoning rollouts.

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: [Yes]

   Justification: Section 5.1 specifies the models (Qwen3-1.7B, Qwen3-4B, Llama-3.2-3B-Instruct), training frameworks (ROLL and VeRL), datasets (DeepMath-103K, DAPO-Math-17K), and key hyperparameters. Appendix C provides the full hyperparameter tables (Tables 6 and 7), window edge handling details (C.2), and software/compute environment (C.3). The PEPO algorithm is fully specified in Eqs. (6)–(8).

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 datasets and base models are cited, but the paper does not provide open access to the PEPO implementation code.

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: [Yes]

Justification: Section 5.1 reports training framework, dataset, group size $G = 8$, learning rate 1×10^{-6}, clipping threshold $\epsilon_h = 0.28$, 500 training steps, and window size $W = 101$. Appendix C, Tables 6 and 7, provide the complete set of hyperparameters including optimizer (AdamW), sampling temperature, top-p, and maximum response length for both the main and single-stream experiments.

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: [No]

   Justification: Main results are averaged over three runs. Tables 2 and 3 report the mean and sample SD for all methods, including Base, and Appendix Tables 13 and 14 provide the per-run results. The paper does not report inferential hypothesis tests.

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: [Yes]

   Justification: Appendix C.3 specifies that all experiments are run on 2 NVIDIA H200 GPUs using ROLL and VeRL frameworks with vLLM for rollout generation. Appendix D (Table 8) reports the wall-clock training time for each method on Qwen3-1.7B and Qwen3-4B, demonstrating that PEPO adds negligible overhead over GRPO.

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 research uses publicly available open-source models and mathematical reasoning datasets, involves no human subjects, and does not collect or release any personal data. The work conforms to the NeurIPS Code of Ethics in all respects.

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 both potential positive impacts (reduced training compute and barrier to entry for reasoning research) and negative considerations (the dual-use nature of improved LLM reasoning capabilities) of the work.

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: The paper does not release new pretrained models, image generators, or scraped datasets.

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: [Yes]

    Justification: All assets used are properly cited: base models, datasets, and frameworks (ROLL, VeRL). All assets are used in accordance with their respective licenses for academic research.

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: The paper does not release new datasets or models. The PEPO method is specified in Section 4 and the training settings are provided in Appendix C.

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 paper does not involve crowdsourcing or research with human subjects.

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 paper does not involve human subjects research and therefore does not require IRB approval.

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: [N/A]

    Justification: LLMs (Qwen3-1.7B, Qwen3-4B, Llama-3.2-3B-Instruct) are the subject of study rather than a tool used in the methodology development. LLMs were not used as a non-standard component of the core method beyond the standard writing/editing assistance for which declaration is not required.