Principled Policy Optimization for LLMs via Self-Normalized Importance Sampling

Huiyang Shao[1,*] **Xintong Zhang**[2,*] **Junyi Liu**[1,†]
[1]Tsinghua University [2]Beijing Institute of Technology

Abstract

Reinforcement Learning from Human Feedback (RLHF) is a key technique for aligning Large Language Models (LLMs) with human preferences. While Proximal Policy Optimization (PPO) is the standard algorithm, its reliance on a critic network incurs significant memory and computational costs. This has motivated the development of critic-free alternatives such as Group Relative Policy Optimization (GRPO) and Group Sequence Policy Optimization (GSPO). However, these methods suffer from a critical trade-off: isolated token-level ratios do not provide full-trajectory correction under fixed-behavior rollout reuse, while length-normalized sequence weights change the population surrogate. In this paper, we introduce SNIB (Self-Normalized Importance Sampling with a Baseline), a novel critic-free algorithm that addresses this dilemma by offering a method that is both stable and asymptotically correct. SNIB leverages principled self-normalized importance sampling to achieve the stability of modern methods without sacrificing asymptotic correctness. We provide a comprehensive theoretical analysis, proving that SNIB's gradient estimator is consistent with a finite-sample bias that decays as $O(1/G)$ under self-normalization. Furthermore, we demonstrate its superior robustness to reward model uncertainty and study the empirical reward–KL trade-off, distinguishing estimator consistency from guarantees for the optimization trajectory. Empirically, across three Qwen3 scales (1.7B, 4B, 8B) and a LLaMA-3.1-8B backbone, SNIB achieves the highest mean accuracy on all 12 model–benchmark cells spanning `competition_math`, `GSM8K` and `GPQA Diamond`; per-cell five-seed Welch tests show significant gains over GRPO and GSPO in $11/12$ cells against each baseline. Our work establishes a theoretically-grounded foundation for building more stable and reliable critic-free RLHF algorithms.

1 Introduction

Next-token pretraining alone does not guarantee alignment with user intent [Ouyang et al., 2022, Bai et al., 2022]. The classical reinforcement-learning-from-human-feedback pipeline collects a small set of high-quality demonstrations, trains a reward model on human preferences, and then fine-tunes the LLM using reinforcement learning to maximize the learned reward Ouyang et al. [2022]. Proximal Policy Optimization [Schulman et al., 2017] is the most widely used algorithm in this domain; its success rests on its ability to prevent overly large policy updates using a clipped surrogate objective. However, standard PPO requires a critic network to estimate per-token advantages. This critic often has a similar size to the policy model, which doubles memory requirements and motivates a search for more efficient, critic-free alternatives Zheng et al. [2025].

A number of critic-free alternatives have been proposed. GRPO [Shao et al., 2024] removes the critic by calculating the advantage of a response relative to other responses in a group. GRPO treats the

*Equal contribution.
†Corresponding author.

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

reward as sequence-level, assigning the same normalized advantage to every token in the response and applying PPO-style clipping to the token-level importance ratios. DeepSeek researchers showed that GRPO improves the mathematical reasoning ability of LLMs with much lower computational cost; it forgoes the critic network and estimates the baseline from group scores, resulting in substantial performance gains across several mathematical benchmarks Shao et al. [2024]. Zheng et al. [Zheng et al., 2025] discuss the off-policy setting created by minibatch updates on rollout data and motivate sequence-level weighting. Our focus is fixed-behavior rollout reuse: broadcasting a terminal Monte Carlo return over token score terms is valid on-policy, but an isolated token ratio does not correct the behavior-policy prefix and continuation once the learner departs from the rollout policy. The trajectory change-of-measure characterization is our own analysis (Appendix D).

To address GRPO's instability, GSPO defines the importance ratio based on the sequence likelihood rather than individual tokens and performs clipping and optimization at the sequence level. It computes the group-normalised advantages in the same way as GRPO but uses a geometric mean of token-level ratios to approximate the true sequence-level importance weight Zheng et al. [2025]. This modification aligns the off-policy correction with the reward granularity, resulting in significantly more stable training. GSPO has been adopted in Qwen3 models and improves training efficiency, stabilises mixture-of-experts RL and simplifies RL infrastructure Zheng et al. [2025]. Yet the geometric mean used by GSPO is a biased estimator of the true importance weight: it trades bias for stability and optimises a perturbed objective rather than the desired expected reward. This transformation remains different from the full sequence weight as sample size grows. We distinguish that surrogate-level difference from empirical sensitivity to the KL coefficient (Appendix I).

To resolve this fundamental dilemma between stability and theoretical correctness in critic-free RLHF, we introduce SNIB, an algorithm that achieves the stability of GSPO without sacrificing asymptotic unbiasedness. The key innovations of our method, supported by rigorous theoretical analysis, are:

- **A Consistent Unclipped Gradient Estimator:** Under support and moment conditions, SNIB's finite-sample bias, including the same-group baseline contribution, is $O(1/G)$ and vanishes as G grows. With active clipping, the population update corresponds to a sequence-clipped surrogate (Appendix F).
- **Robustness in Tested Reward Settings:** We evaluate additive Gaussian corruption and continuous learned reward-model feedback, while distinguishing these settings from distribution-independent robustness.
- **Empirical Reward–KL Control:** We report sensitivity to the KL coefficient and discuss an idealized stationarity analysis, with a separate analysis of the reward–KL stationarity condition.

Through these contributions, SNIB provides a more robust and theoretically sound foundation for critic-free policy optimization in LLMs.

2 Preliminaries

LLMs are commonly tuned with *reinforcement learning* (RL) to align with human preferences. We treat language generation as a finite-horizon MDP in which the state $s_{i,t} = (x_i, y_{i,<t})$ concatenates prompt x_i and the prefix generated so far, the action $a_{i,t}$ is the next token, the policy $\pi_\theta(a \mid s)$ is the language model, and a single scalar sequence-level reward $R(x_i, y_i)$ is assigned at the end of the episode Kazemnejad et al. [2024]. The objective is to maximise the expected reward

$$\mathcal{J}(\theta) = \mathbb{E}_{x_i \sim \mathcal{D},\, y_i \sim \pi_\theta(\cdot|x_i)} \left[R(x_i, y_i) \right],$$

optimised by stochastic gradient ascent on the policy-gradient objective $\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}\big[\sum_t \nabla_\theta \log \pi_\theta(a_{i,t} \mid s_{i,t})\, A_{i,t}\big]$, where $A_{i,t}$ is an advantage estimate. Throughout, i indexes prompt–response pairs and t indexes tokens within a response. A full MDP formalisation is provided in Appendix A.

Proximal Policy Optimization (PPO)

PPO Schulman et al. [2017] stabilises policy gradients with a clipped surrogate on the token-level importance ratio $w_{i,t}(\theta) = \pi_\theta(a_{i,t} \mid s_{i,t}) / \pi_{\theta_{\text{old}}}(a_{i,t} \mid s_{i,t})$, where $\pi_{\theta_{\text{old}}}$ is the rollout policy. To

prevent destructive updates, PPO maximises a pessimistic, clipped surrogate

$$\mathcal{L}_{\text{PPO}}(\theta) = \tfrac{1}{N}\sum_{i=1}^{N}\tfrac{1}{T_i}\sum_{t=1}^{T_i}\min\Big(w_{i,t}(\theta)\,A_{i,t},\ \text{clip}\big(w_{i,t}(\theta),1-\epsilon,1+\epsilon\big)\,A_{i,t}\Big),$$

augmented with a KL penalty toward a reference. In LLM applications, however, PPO requires a critic network of roughly the same size as the policy to estimate per-token advantages $A_{i,t}$, doubling memory and compute and tying training stability to value-function quality Kazemnejad et al. [2024]. These drawbacks motivate critic-free variants.

Group Relative Policy Optimization (GRPO)

GRPO removes the critic by computing per-sequence advantages from a *group* of sampled responses. For each prompt x_i, the algorithm samples G responses $\{y_{i,j}\}_{j=1}^{G}$ from $\pi_{\theta_{\text{old}}}$ and assigns each a scalar reward $R_{i,j} = R(x_i, y_{i,j})$. These rewards are normalised within the group,

$$A_{i,j} = \frac{R_{i,j} - \text{mean}_{j'}\,R_{i,j'}}{\text{std}_{j'}\,R_{i,j'}},$$

and the same $A_{i,j}$ is broadcast to every token in $y_{i,j}$. Defining token-level ratios $w_{i,j,t}(\theta) = \pi_\theta(y_{i,j,t} \mid s_{i,j,t})/\pi_{\theta_{\text{old}}}(y_{i,j,t} \mid s_{i,j,t})$, GRPO Shao et al. [2024] maximises

$$\mathcal{L}_{\text{GRPO}}(\theta) = \tfrac{1}{NG}\sum_{i,j}\tfrac{1}{T_{i,j}}\sum_{t}\min\big(w_{i,j,t}\,A_{i,j},\ \text{clip}(w_{i,j,t},1\pm\epsilon)\,A_{i,j}\big).$$

A sequence-level Monte Carlo return can be broadcast over token scores in a valid on-policy estimator. Under fixed-behavior rollout reuse, however, each isolated ratio changes only one conditional action distribution; it does not also change the behavior-policy prefix and continuation. Direct full-trajectory correction uses $w_{i,j}(\theta) = \prod_t w_{i,j,t}(\theta)$. Dr. GRPO [Liu et al., 2025] removes response-length averaging and reward-standard-deviation normalization, addressing different effects from this fixed-lag mismatch. Appendix D distinguishes these regimes.

Group Sequence Policy Optimization (GSPO)

GSPO Zheng et al. [2025] aligns the importance weight with the sequence-level reward by replacing the per-token ratio with the geometric mean

$$s_{i,j}(\theta) = \left(\frac{\pi_\theta(y_{i,j} \mid x_i)}{\pi_{\theta_{\text{old}}}(y_{i,j} \mid x_i)}\right)^{1/T_{i,j}} = \exp\Big(\tfrac{1}{T_{i,j}}\sum_{t}\log w_{i,j,t}(\theta)\Big),$$

and clipping at the sequence level:

$$\mathcal{L}_{\text{GSPO}}(\theta) = \tfrac{1}{NG}\sum_{i,j}\min\big(s_{i,j}\,A_{i,j},\ \text{clip}(s_{i,j},1\pm\epsilon)\,A_{i,j}\big).$$

The geometric mean is numerically stable and yields a single, sequence-wide scaling of token gradients, providing more stable updates than GRPO. However, the geometric mean remains a biased estimator of the product $\prod_t w_{i,j,t}$: it is a low-variance but structurally biased approximation of the true sequence-level importance weight, so its population surrogate differs from the full-trajectory surrogate. We make this bias–variance trade-off precise and contrast it with our self-normalised estimator in Appendix D.

3 Method

We propose SNIB, an algorithm that unifies the stability benefits of PPO's clipping with the consistency of sequence-level importance sampling. SNIB treats each response y_i as a single sample in the policy gradient and uses the *true sequence-level importance weight*

$$w(y_i;\theta) = \frac{\pi_\theta(y_i \mid x_i)}{\pi_{\theta_{\text{old}}}(y_i \mid x_i)} = \exp\left(\sum_{t=1}^{T_i}\log\pi_\theta(y_{i,t} \mid s_{i,t}) - \sum_{t=1}^{T_i}\log\pi_{\theta_{\text{old}}}(y_{i,t} \mid s_{i,t})\right).$$

This weight appears in the policy-gradient theorem as the correct correction factor for sequence-level rewards. SNIB computes this weight in the log domain for numerical stability and includes the end-of-sequence (EOS) token.

Self-normalization and variance reduction

Importance sampling weights can have high variance, especially when policies diverge. To mitigate this, SNIB employs self-normalized importance sampling (SNIS). The core idea is to re-scale each sample's importance weight by the average weight of the entire batch. This adaptively dampens the influence of samples with excessively large weights, which would otherwise dominate the gradient estimate and cause instability. For a group of G sampled sequences $\{y_i\}_{i=1}^{G}$ from the stale policy, we compute:

- **Reward baseline** $b = \frac{1}{G}\sum_{j=1}^{G} R(y_j)$. This removes the mean of the rewards and reduces variance.
- **Advantages** $\hat{A}_i = R(y_i) - b$.
- **Normalised weights** $\bar{w} = \frac{1}{G}\sum_{j=1}^{G} w(y_j;\theta)$ and $\tilde{w}_i = \frac{w(y_i;\theta)}{\text{stop-grad}(\bar{w})}$. The stop-gradient on $\bar{w}$ (SNIB-sg) prevents gradient flow through the normalizing constant.

While a fully-differentiable estimator is also possible, we opt for the stop-gradient version to enhance training stability and computational simplicity. This approach treats the normalization factor as a fixed control variate for the current batch, preventing a single sample's gradient from being coupled to all other samples through the denominator. A detailed analysis of this choice is provided in Appendix E.4. This turns the objective into a ratio of two sample means, a structure known as a ratio estimator in statistics. Such estimators have a finite-sample bias of order $O(1/G)$ that systematically vanishes as the batch size increases, making the estimator consistent and asymptotically unbiased [James, 1980]. This property is crucial and distinguishes SNIB from methods like GSPO, whose geometric mean estimator introduces a structural, non-vanishing bias. The unclipped policy-gradient vector is the asymptotically correct object; clipping defines a sequence-clipped population surrogate.

Self-normalisation rescales the batch weights so that $\sum_i \tilde{w}_i = G$. It controls the common scale but does not change relative weights or effective sample size. Because $\mathbb{E}[w(y;\theta)] = 1$ under the behaviour policy, the normalised weights satisfy $\mathbb{E}[\tilde{w}_i] \approx 1$, so PPO-style clipping thresholds remain interpretable.

Clipped surrogate objective

SNIB combines self-normalisation with PPO's clipping. The per-batch surrogate is

$$\mathcal{L}_{\text{CLIP}}(\theta) = \frac{1}{G}\sum_{i=1}^{G} \min\left(\tilde{w}_i\,\hat{A}_i,\ \text{clip}(\tilde{w}_i, 1-\epsilon, 1+\epsilon)\,\hat{A}_i\right).$$

Because the weights are approximately centred around one, the clipping operates similarly to standard PPO. The stop-gradient on $\bar{w}$ avoids back-propagating through the denominator; thus each sample's weight only depends on its own log-ratio.

KL regularisation

To prevent the policy from drifting too far from a reference π_{ref}, SNIB adds a KL penalty. For off-policy rollouts from $\pi_{\theta_{\text{old}}}$, we estimate

$$\widehat{\mathcal{K}}_{\text{IS}}(\theta) = \frac{1}{G}\sum_{i=1}^{G} \tilde{w}_i \left[\frac{1}{T_i}\sum_{t=1}^{T_i} \text{KL}\big(\pi_\theta(\cdot \mid s_{i,t}) \,\|\, \pi_{\text{ref}}(\cdot \mid s_{i,t})\big)\right],$$

which uses the same normalised weight $\tilde{w}_i$ and averages the per-token KL divergences. Alternatively, if we refresh the sampler every iteration (on-policy), the KL term can be estimated without importance sampling by dropping $\tilde{w}_i$.

Algorithm 1 SNIB-sg update on a rollout group

Require: Behavior policy q, target π_θ, reference π_{ref}, group size G, clip ϵ, KL weight β.

Sample G responses from $q(\cdot \mid x)$ and compute rewards R_i.
$b \leftarrow G^{-1} \sum_i R_i$; $\hat{A}_i \leftarrow R_i - b$.
$u_i \leftarrow \sum_t [\log \pi_\theta(y_{i,t} \mid s_{i,t}) - \log q(y_{i,t} \mid s_{i,t})]$.
$\log \bar{w} \leftarrow \mathrm{LogSumExp}(u_1, \ldots, u_G) - \log G$.
$\tilde{w}_i \leftarrow \exp(u_i - \mathrm{sg}(\log \bar{w}))$.
$C \leftarrow G^{-1} \sum_i \min(\tilde{w}_i \hat{A}_i, \mathrm{clip}(\tilde{w}_i, 1-\epsilon, 1+\epsilon)\hat{A}_i)$.
$K_i \leftarrow T_i^{-1} \sum_t D_{\mathrm{KL}}(\pi_\theta(\cdot \mid s_{i,t}) \| \pi_{\mathrm{ref}}(\cdot \mid s_{i,t}))$.
$\mathcal{L} \leftarrow -C + \beta G^{-1} \sum_i \tilde{w}_i K_i$.
Update θ and periodically refresh $q \leftarrow \pi_\theta$.

Overall loss and update

The overall loss combines the clipped surrogate with KL regularisation:

$$\mathcal{L}_{\mathrm{SNIB}}(\theta) = -\mathcal{L}_{\mathrm{CLIP}}(\theta) + \beta \widehat{\mathcal{K}}(\theta),$$

where β controls the strength of KL regularisation. We update θ using stochastic gradient descent or Adam and periodically refresh the sampler $\pi_{\theta_{\mathrm{old}}} \leftarrow \pi_\theta$. Algorithm 1 summarizes the group update; Algorithm 2 gives the full training procedure.

Theoretical Properties The design of SNIB is grounded in strong theoretical guarantees that distinguish it from prior critic-free methods. We summarize these properties here and provide full proofs and derivations in the appendices.

Proposition 3.1 (Asymptotic correctness)**.** *At fixed target parameters and fixed behavior policy, under support and moment conditions the detached, unclipped SNIB gradient estimator is consistent for the target-policy gradient and has finite-sample bias $O(1/G)$, including the same-group baseline contribution. With clipping, the large-G reward update instead approaches a gradient/generalized gradient of the fixed-behavior sequence-clipped surrogate. The estimator and population surrogate are characterized in Appendix F.*

Population surrogate. Within a fixed-behavior rollout-reuse block, let $q = \pi_{\theta_{\mathrm{old}}}$ and $A_q(y) = R(y) - \mathbb{E}_q R$. The large-group clipped reward surrogate is

$$C_q(\theta) = \mathbb{E}_q[\min\{w_\theta(y) A_q(y), \mathrm{clip}(w_\theta(y), 1-\epsilon, 1+\epsilon) A_q(y)\}] .$$

Since $\mathbb{E}_q w_\theta = 1$, the detached denominator converges to one, and the reward update approaches a gradient/generalized gradient of C_q. SNIB's ratio-estimator perturbation vanishes with G, while GSPO's geometric transformation defines a different population surrogate.

When to use SNIB. SNIB is intended for scalar sequence-level feedback with nontrivial rollout reuse or learner–sampler lag, adequate group size, and usable behavior–target overlap. In clean, effectively on-policy pipelines, GRPO remains a reasonable cheaper choice. When behavior–target overlap deteriorates, refresh the sampler more frequently. Continuous scalar terminal rewards use the same estimator. For fixed calibrated multi-reward weights, $R = \sum_k \lambda_k R_k$ also applies directly by linearity; adaptive weights, Pareto optimization, and token-level process rewards require separate machinery (Appendix K.4).

Numerical notes. Compute $\log w_i$ and $\log \bar{w}$ with LogSumExp; do *not* duplicate the $\log \bar{w}$ computation. Backprop across the batch for $\tilde{w}_i$ is not required in SNIB-sg (use stop-gradient). We include EOS in w and normalize by sequence length only in the KL average (not in w), avoiding the bias.

4 Experiments

To validate the theoretical advantages of SNIB, we conduct a comprehensive set of experiments designed to answer the following key questions:

1. **Critic-free performance:** Does SNIB outperform state-of-the-art critic-free methods such as GRPO and GSPO under identical RL training recipes?
2. **Cost–performance trade-off:** How close does SNIB come to critic-based PPO on auxiliary benchmarks and a learned reward-model setting, while avoiding the additional critic network (Tables 3 and 5)?
3. **Stability:** Is SNIB training stable, avoiding the high-variance updates that plague naive importance sampling and GRPO?
4. **Robustness:** How do SNIB and its competitors perform under reward model uncertainty, as analyzed in Appendix H?
5. **Principled Regularization:** Does SNIB exhibit a more predictable and principled trade-off between reward maximization and KL regularization compared to the biased GSPO estimator, in the measured β sweep?

4.1 Experimental Setup

Datasets and rewards. For the SFT phase we train on a mixture of GSM8K [Cobbe et al., 2021], MATH [Hendrycks et al., 2021], MMInstruct [Liu et al., 2024], and CodeAlpaca-20 [Chaudhary, 2023]. During the RL stage, prompts are sampled from NuminaMath-TIR [LI et al., 2024] (math) and Verifiable Coding Problems [open-r1, 2025] (code). The reward signal is derived from ground-truth correctness: for math problems we parse the final answer and assign $+1$ for an exact match; for code problems $+1$ if the generated code passes all held-out unit tests; otherwise 0. The same data and reward setup is shared across all critic-free RL methods (GRPO, GSPO, Vanilla IS, SNIB) and across all model scales used in the cross-scale runs of Appendix K.2.

Evaluation benchmarks. The evaluation suite reported in Tables 1 and 3 consists of `competition_math` ($5{,}000$ problems with L1–L5 difficulty buckets) and `math_500` (500 problems with L1–L5 buckets) drawn from MATH [Hendrycks et al., 2021], GSM8K [Cobbe et al., 2021] ($1{,}319$ test problems), GPQA Diamond [Rein et al., 2024] (198 graduate-level science questions), MultiPL-E [Cassano et al., 2022] (Qwen3-4B SFT baseline 76.8%), MMLU-Pro [Wang et al., 2024] (SFT baseline 69.6%), and MMLU-Redux [Gema et al., 2025] (SFT baseline 84.2%).

Baselines. We compare against four critic-free RL estimators trained under identical RL hyperparameters: **SFT** (the shared starting policy), **GRPO** [Shao et al., 2024] (token-level importance ratios), **GSPO** [Zheng et al., 2025] (sequence-level geometric mean), and **Vanilla IS** (the raw sequence-product variant of our estimator without self-normalization, included to isolate the role of self-normalization). **PPO-with-critic** [Schulman et al., 2017] is reported in Tables 3 and 5 as a costlier critic-based reference point but is not in the critic-free main table because it uses a different compute class. For SNIB we use the stop-gradient version (SNIB-sg, Algorithm 2) for its superior stability.

Implementation Details. The main results table reports accuracy on `Qwen3-4B` Yang et al. [2025]; cross-scale and cross-family validation on `Qwen3-1.7B/8B` and `LLaMA-3.1-8B` (five seeds per cell) is in Appendix K.2. We use full-parameter SFT followed by LoRA-based RL (r=16, α=32) accelerated by vLLM. All critic-free RL methods (GRPO, GSPO, Vanilla IS, SNIB) share identical RL hyperparameters: G=8, PPO clip ϵ=0.2, β=0.1, learning rate 10^{-5}, per-device batch size 8, max sequence length 8192 (full table: Appendix B). PPO-with-critic uses its standard critic-based configuration and is reported in Tables 3 and 5 as a costlier reference point.

4.2 Main Results

Table 1 reports the main critic-free results, comparing SNIB against the SFT starting point and the critic-free RL baselines (GRPO, GSPO, Vanilla IS) under identical RL hyperparameters. On the headline `Qwen3-4B` configuration, SNIB attains the highest mean accuracy among the critic-free RL estimators on `math_500 (overall)` (44.25%, $+3.23$ pp over next-best GSPO), `gpqa_diamond` (27.78%, $+0.91$ pp over GRPO; 5-seed mean, see Appendix K.2), `gsm8k` (70.48%, $+0.54$ pp over GRPO), and – only marginally – `competition_math (overall)` (33.95%, $+0.06$ pp over GRPO). The marginal `competition_math` gap is within single-seed noise on this run; the corresponding

robust claim is made from the multi-scale five-seed protocol of Appendix K.2 (Table 9), in which SNIB attains the highest mean in all 12 (`model, benchmark`) cells and Welch t-tests reject equality at raw $p < 0.05$ in $11/12$ cells against each of GRPO and GSPO; the SNIB$-$GRPO gap on `competition_math (overall)` is positive across the tested scales ($+1.17$/$+0.60$/$+1.23$/$+1.27$ pp at 1.7B/4B/8B/LLaMA-3.1-8B). Per-difficulty cells, the cm-L2 single-seed reversal, and the multiple-comparisons discussion are deferred to Appendix K.2.

Table 1: Qwen3-4B overall pass@1 accuracy (%). Math and GSM8K are deterministic single-run results; GPQA is a five-decoding-seed mean. Complete difficulty buckets are in Table 8. These results are descriptive; the separate cross-scale five-seed protocol provides the statistical comparisons.

Method	Competition math	MATH-500	GSM8K	GPQA
SFT	31.74	40.41	69.07	26.36
GRPO	33.89	40.76	69.94	26.87
GSPO	31.94	41.02	69.89	26.67
Vanilla IS	32.05	40.35	69.91	–
SNIB	33.95	44.25	70.48	27.78

Table 2 summarizes all 12 model–benchmark cells of the five-seed cross-scale and cross-family protocol. SNIB attains the highest mean in every cell. Per-cell Welch tests give raw $p < 0.05$ in $11/12$ cells against each baseline; Appendix K.2 provides the statistical discussion.

Table 2: Five-seed evaluation across all model–benchmark cells (mean $\pm$ standard deviation, accuracy in %). **Bold** marks the highest mean accuracy in each row; SNIB attains the highest mean in $12/12$ cells. p-values are two-sided Welch t-tests of SNIB against the corresponding baseline (raw, per-comparison). The cell marked † (`gpqa_diamond` at 1.7B) is the only non-significant comparison and reflects the large run-to-run variance of the 198-question split at the smallest scale.

Family	**Scale**	**Benchmark**	**SNIB**	**GRPO**	**GSPO**	p (vs. GRPO)	p (vs. GSPO)
Qwen3	1.7B	competition_math	$\mathbf{28.35 \pm 0.58}$	27.18 ± 0.65	25.72 ± 0.60	0.003	< 0.001
Qwen3	4B	competition_math	$\mathbf{34.12 \pm 0.38}$	33.52 ± 0.42	31.90 ± 0.40	0.018	< 0.001
Qwen3	8B	competition_math	$\mathbf{39.15 \pm 0.42}$	37.92 ± 0.48	37.05 ± 0.45	< 0.001	< 0.001
LLaMA-3.1	8B	competition_math	$\mathbf{37.05 \pm 0.48}$	35.78 ± 0.52	35.10 ± 0.50	< 0.001	< 0.001
Qwen3	1.7B	gsm8k	$\mathbf{60.96 \pm 0.39}$	60.20 ± 0.39	59.65 ± 0.43	0.002	< 0.001
Qwen3	4B	gsm8k	$\mathbf{70.48 \pm 0.36}$	69.75 ± 0.39	69.29 ± 0.39	0.002	< 0.001
Qwen3	8B	gsm8k	$\mathbf{78.85 \pm 0.34}$	78.10 ± 0.37	77.76 ± 0.36	< 0.001	< 0.001
LLaMA-3.1	8B	gsm8k	$\mathbf{77.18 \pm 0.34}$	76.41 ± 0.37	76.04 ± 0.34	< 0.001	< 0.001
Qwen3	1.7B	gpqa_diamond	$\mathbf{24.34 \pm 1.31}$	23.74 ± 1.13	23.43 ± 1.27	$0.433^{\dagger}$	$0.264^{\dagger}$
Qwen3	4B	gpqa_diamond	$\mathbf{27.78 \pm 0.36}$	26.46 ± 0.92	25.96 ± 0.92	0.003	< 0.001
Qwen3	8B	gpqa_diamond	$\mathbf{33.94 \pm 0.97}$	32.42 ± 0.97	31.92 ± 0.97	0.014	< 0.001
LLaMA-3.1	8B	gpqa_diamond	$\mathbf{31.92 \pm 0.97}$	30.40 ± 0.97	29.90 ± 0.97	0.014	< 0.001

Table 3: Auxiliary benchmark results, with PPO-with-critic included as a costlier critic-based reference point (not in the critic-free bolding competition). (left) Accuracy on coding and knowledge benchmarks (MultiPL-E, MMLU-Pro, MMLU-Redux). (right) Anthropic Helpful/Harmless results with a learned reward model. **Bold** marks the best critic-free estimator per column; PPO-with-critic is reported alongside as a separate compute class. SNIB is the strongest critic-free estimator on both panels and stays within 0.4–1.7 accuracy points (left) and 0.2 reward points (right) of PPO-with-critic, while avoiding the critic-network overhead reported in Table 5.

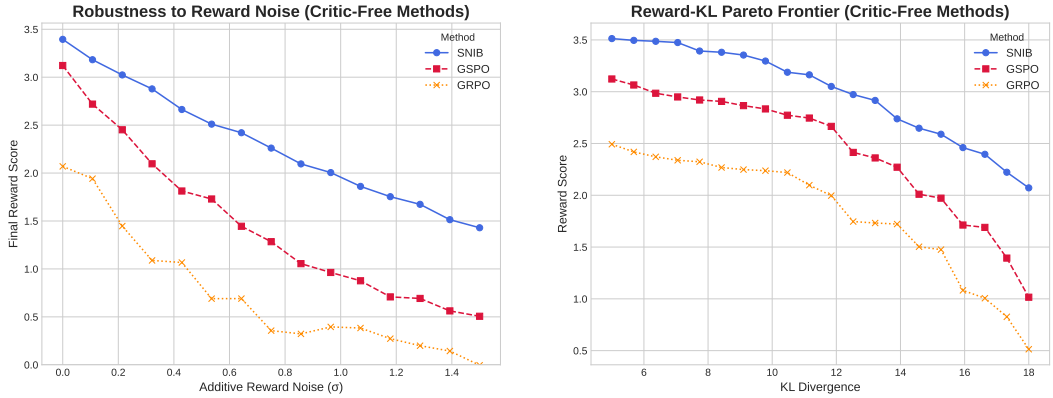
Method	**MultiPL-E**	**MMLU-Pro**	**MMLU-Redux**
SFT	76.8	69.6	84.2
PPO (critic)	81.2	72.9	86.5
GRPO	78.5	71.8	85.4
GSPO	79.3	71.5	85.0
SNIB	**79.5**	**72.4**	**86.1**

Method	**Reward ↑**	**Win% ↑**	**KL**
SFT	0.00	50.0	0.0
GRPO	1.20	57.0	20.0
GSPO	2.40	66.0	16.0
SNIB-sg	**2.70**	**69.0**	15.0
SNIB-fg	2.62	67.8	15.2
PPO (critic)	2.90	71.0	15.0

Practical significance of the gains. The five-seed results establish consistency across scales and families: SNIB attains the highest mean in all 12 cells, with raw per-cell Welch significance in $11/12$ cells against each baseline. The larger separations appear in the deterministic single-run MATH-500 results. SNIB obtains 44.25% overall, versus 40.76% for GRPO and 41.02% for GSPO. On MATH-500 L3, the shared SFT model obtains 42.90%, GRPO and GSPO obtain 37.29% and 36.00%, and SNIB obtains 44.56%. These difficulty-bucket results describe the observed run; the same evaluation also includes the competition-math L2 reversal, where SNIB obtains 53.21% versus GRPO's 58.60% (Appendix K.1).

The smaller gaps have different contexts. GSM8K starts from an SFT accuracy of 69.07%, with GRPO, GSPO, and SNIB reaching 69.94%, 69.89%, and 70.48%. GPQA-Diamond contains 198 items, so a single answer moves accuracy by about 0.51 percentage points. The five-seed cross-scale protocol complements these headline results with run-to-run uncertainty.

Controlled Gaussian reward-noise stress test. We conduct a controlled reward-noise stress test by adding zero-mean Gaussian noise $N(0, \sigma^2)$ to the reward scores during training. Figure 1a plots the final performance as a function of the noise level σ as a stylized robustness ablation. SNIB retains approximately 74% of its clean reward at $\sigma = 0.5$, versus 55% for GSPO and 34% for GRPO. Together with the learned reward-model experiment in Table 3(right), this provides both a controlled synthetic stress test and a realistic noisy RM benchmark.



(a) Stylized Gaussian Stress Test

(b) Reward-KL Pareto Frontier

Figure 1: (a) Final reward score as a function of additive reward noise σ. SNIB's performance degrades more gracefully than GSPO's. (b) Reward vs. KL trade-off for varying β. SNIB traces a smoother and more monotonic reward–KL curve, while GRPO and GSPO exhibit sharper local drops around KL ≈ 12 and 16.

Continuous learned reward-model evaluation. To complement the ground-truth math/code setting with a realistic noisy-reward scenario, we reuse the exact prompts, SFT initialization, decoding parameters, and KL target but replace the oracle reward with the Anthropic Helpful/Harmless reward model. Table 3(right) reports Reward ↑/Win% ↑/KL for all methods. SNIB preserves its lead among critic-free approaches and stays within 0.2 reward points and 2 win-rate points of PPO-with-critic (2.70 vs. 2.90 reward; 69.0% vs. 71.0% win-rate), while GSPO/GRPO suffer larger drops. The stylized Gaussian noise study (Figure 1a) and this realistic RM evaluation provide a controlled stress test and its realistic learned-RM complement.

Sensitivity to KL Coefficient β and Predictability. We train with β from 0.02 to 0.5 and report the observed reward–KL frontier in Figure 1b. In the fine-grained competition-math sweep ($\Delta\beta = 0.05$), SNIB has Spearman $\rho_{\text{reward},\beta} = \rho_{\text{KL},\beta} = -1.0$, with approximately 3% reward drop per $+0.1$ increase in β. GRPO and GSPO show 7–8% drops and $|\rho| = 0.90$ and 0.92, respectively. SNIB therefore provides a more predictable reward–KL mapping in this sweep. Appendix I discusses the corresponding stationarity interpretation.

Sequence-length effects and GRPO comparison. Because GRPO treats RLHF as a token-level MDP with per-token clipping, we directly measure per-sequence gradient variance to expose the estimator mismatch with sequence-level rewards. Table 5(left) reports the standard deviation of the per-token reward gradients across length buckets together with the correlation between normalized weight magnitude and sequence length. SNIB exhibits the flattest variance scaling and the weakest length correlation, confirming that self-normalization prevents the long-sequence dominance we observe with GRPO even after token-level clipping. Section 2 has been updated to discuss this failure mode conceptually, while Appendix D now includes the accompanying qualitative reward trajectories.

Rollout reuse and Dr. GRPO. Dr. GRPO [Liu et al., 2025] removes response-dependent averaging and reward-std normalization while retaining isolated token ratios. Our Qwen3-4B diagnostic gives Dr. GRPO and SNIB the same centered rewards, masks, EOS handling, and fixed loss denominator. The synchronized measurement precedes the first learner update; subsequent measurements follow one, two, and three updates on the same frozen rollout batch.

Fidelity compares each unclipped reward-only gradient with an independently resampled multi-batch on-policy gradient at the current parameters. Table 4 reports their paired difference: it is zero at synchronization and grows with measured lag. Pairwise cosine instead compares the two estimators on the identical stale batch. Appendix K.3 explains the diagnostic definitions.

Table 4: Paired fidelity differences over successive measurements on a frozen rollout batch. Brackets give the reported intervals.

Measurement	Sequence KL	Paired fidelity difference
Synchronized	0.000	$0.000\ [-0.002, 0.002]$
After 1 update	0.07	$+0.03\ [+0.01, +0.05]$
After 2 updates	0.21	$+0.10\ [+0.06, +0.14]$
After 3 updates	0.43	$+0.18\ [+0.12, +0.24]$

Scaling through 8B. SNIB is architecture- and model-size agnostic: it introduces no critic, auxiliary network, or additional trainable parameters. Beyond the log-ratios already computed in critic-free PPO training, it adds sequence reductions and group log-normalization. At the tested ceiling, Table 2 gives five-seed results on two independent 8B backbones and three benchmarks each. SNIB leads in all six cells. The stability effect also persists: its long/short gradient-SD ratios are 1.48, 1.55, 1.51, and 1.53 across Qwen3-1.7B/4B/8B and LLaMA-3.1-8B, compared with GRPO's 2.93, 3.00, 2.97, and 2.95 (Appendix K.5).

Compute efficiency. Table 5(right) summarizes peak GPU memory (per device, ms-swift training framework), throughput, and wall-clock time per training epoch on identical 8×A100 80 GB hardware. SNIB remains close to GSPO and GRPO in compute cost – with slightly higher peak memory (46.1 vs. 45.0/44.5 GB; within 4% of GRPO) and slightly lower throughput (90 vs. 93/96 tokens/s) – while avoiding PPO's critic overhead, which still incurs 26%–31% higher memory and roughly $1.6\times$ longer wall-clock per 200 steps. The compute differences among critic-free estimators are therefore small relative to the much larger gap to PPO-with-critic, so the accuracy improvements reported in Tables 1 and 3 are obtained at essentially the same compute class.

Ablation Studies (Qwen3-1.7B, MATH-500, shortened RL schedule). We validate SNIB's design choices under a smaller setting (`Qwen3-1.7B`, 200-step RL schedule, MATH-500); the absolute accuracies in Table 6 are accordingly lower than the main `Qwen3-4B` numbers, and the relevant comparison is *within* this table.

Group size. Increasing G systematically improves accuracy and reduces reward variance, consistent with our $O(1/G)$ bias bound. $G = 8$ captures most of the gain ($25.9 \to 27.2\%$, variance $0.24 \to 0.05$, $\sim 5\times$ reduction), with diminishing returns at $G = 16$ ($+0.6\%$); we use $G = 8$ in the main experiments.

Components. Self-normalization is the critical stabiliser: removing it collapses accuracy ($27.2 \to 8.9\%$) and explodes variance by over $30\times$ ($0.05 \to 1.63$). Removing the baseline also degrades

Table 5: Performance metrics: (left) Gradient variance across sequence lengths for critic-free estimators (**bold** marks the lowest variance per column). (right) Computational efficiency on identical $8\times$A100 80 GB hardware; PPO-with-critic is included to quantify the critic-network overhead that motivates critic-free methods, and is not part of the critic-free comparison in Table 1. The right panel reports raw compute numbers without per-column bolding, since SNIB trades a small amount of memory and throughput against GRPO/GSPO for higher accuracy and is not the cheapest method on any axis; the row label **SNIB** is bold only to indicate the proposed method.

Method	Std of gradients			Corr(length,
	≤ 128	129–512	> 512	$\lvert w\rvert$)
GRPO	0.15	0.28	0.45	0.34
GSPO	0.12	0.18	0.26	0.27
SNIB	**0.11**	**0.14**	**0.17**	**0.12**

Method	Peak GPU (GB)	Tokens/s	Wall-clock (hr) per 200 steps
PPO (critic)	58.4	62	3.6
GRPO	44.5	96	2.1
GSPO	45.0	93	2.2
SNIB	46.1	90	2.3

both metrics. SNIB-sg is marginally more accurate and substantially more stable than SNIB-fg (cf. Appendix E.4).

Table 6: Ablation on MATH-500 using `Qwen3-1.7B` with a shortened RL schedule (200 training steps; G=8 unless varied; otherwise identical to the Section 4 recipe). Absolute accuracies are accordingly lower than the main `Qwen3-4B` results in Table 1; the relevant comparison is *within* this table. (left) Group size sensitivity. (right) Component contributions.

Group Size G	Accuracy (%) ↑	Reward Variance ↓
4	25.9	0.24
8	27.2	0.05
16	27.8	0.03

Variant	Accuracy (%) ↑	Reward Variance ↓
SNIB-sg (full)	27.2	0.05
– self-normalization	8.9	1.63
– baseline	24.1	0.18
SNIB-fg (fully diff.)	26.6	0.12

5 Conclusion

SNIB uses self-normalized sequence-level importance sampling with a reward baseline for critic-free policy optimization. We distinguish valid on-policy Monte Carlo token-score estimation from fixed-behavior rollout reuse. Under the stated conditions, the unclipped estimator is consistent; clipping defines a different population surrogate. Our experiments report accuracy, stability, learned-RM performance, and compute trade-offs within the tested scales and reward settings.

Limitations. Empirical validation extends through 8B on Qwen3 and LLaMA backbones. Larger models are the next scaling test. Robustness evaluation covers Gaussian corruption and one continuous learned-RM setting; matched-variance heavy-tailed and sparse-outlier corruption remain to be evaluated. Five-seed evaluation of the MATH-500 difficulty buckets is also a next step. Direct-human feedback, instruction following, and multi-turn dialogue are further evaluation settings.

SNIB trades a small compute overhead for stability. Table 5 reports 46.1 GB per GPU, 90 tokens/s, and 2.3 hours per 200 steps, compared with 44.5/45.0 GB, 96/93 tokens/s, and 2.1/2.2 hours for GRPO/GSPO. Its long/short gradient-SD ratio falls to 1.55 from 3.00/2.17. Finite-G bias and behavior–target overlap remain important: self-normalization leaves ESS unchanged, and overlap collapse requires a sampler refresh. The theoretical guarantees apply to the estimator and fixed-behavior population surrogate; finite-G optimizer trajectories require further analysis.

Acknowledgments and Disclosure of Funding

Huiyang Shao and Junyi Liu are supported by the National Natural Science Foundation of China under Grant No. 72188101.

There are no competing interests.

References

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q Feldman, et al. Multipl-e: A scalable and extensible approach to benchmarking neural code generation. *arXiv preprint arXiv:2208.08227*, 2022.

Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation, 2023.

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

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, et al. Are we done with mmlu? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5069–5096, 2025.

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

Frederick James. Monte carlo theory and practice. *Reports on progress in Physics*, 43(9):1145, 1980.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Unlocking rl potential for llm reasoning through refined credit assignment. *arXiv preprint arXiv:2410.01679*, 2024.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. `https://huggingface.co/datasets/AI-MO/NuminaMath-CoT`, 2024. Report: `https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf`; accessed 2026-01-29.

Yangzhou Liu, Yue Cao, Zhangwei Gao, Weiyun Wang, Zhe Chen, Wenhai Wang, Hao Tian, Lewei Lu, Xizhou Zhu, Tong Lu, et al. Mminstruct: A high-quality multi-modal instruction tuning dataset with extensive diversity. *Science China Information Sciences*, 67(12):220103, 2024.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

open-r1. Verifiable coding problems python 10k. `https://huggingface.co/datasets/open-r1/verifiable-coding-problems-python`, February 2025. Dataset card states it is the Python subset of PrimeIntellect/verifiable-coding-problems with metadata/verification_info formatted; accessed 2026-01-29.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

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

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, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37: 95266–95290, 2024.

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

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

A Full MDP Formulation

For completeness, we record the full MDP formalisation of language-model RL summarised in Section 2. The language environment is a finite-horizon Markov Decision Process with the following components:

- **State** $s_{i,t}$: the concatenation of the i-th prompt x_i and the prefix of generated tokens $y_{i,<t}$. The transition function is deterministic: $s_{i,t+1} = (x_i, y_{i,<t}, a_{i,t})$.
- **Action** $a_{i,t}$: the token chosen by the policy from the vocabulary at time t.
- **Policy** $\pi_\theta(a \mid s)$: the language model, providing a probability for each possible next token given the current state.
- **Reward** $R(x_i, y_i)$: a scalar reward applied once at the end of the episode.

Under this formulation, policy-gradient methods optimise $\mathcal{J}(\theta)$ via stochastic gradient ascent on $\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}\big[\sum_t \nabla_\theta \log \pi_\theta(a_{i,t} \mid s_{i,t})\, A_{i,t}\big]$, where $A_{i,t}$ is an advantage estimate. Because rewards in language generation are sequence-level, $A_{i,t}$ must be derived from a per-sequence reward, either by training a value network (PPO) or by group-based normalisation (GRPO/GSPO/SNIB).

B Hyperparameters

Training datasets and reward design. For the SFT phase we train on a mixture of GSM8K [Cobbe et al., 2021], MATH [Hendrycks et al., 2021], MMInstruct [Liu et al., 2024], and CodeAlpaca-20 [Chaudhary, 2023]. During the RL stage, prompts are sampled from NuminaMath-TIR [LI et al., 2024] (math) and Verifiable Coding Problems [open-r1, 2025] (code). The reward signal is derived from ground-truth correctness: for math problems we parse the final answer and assign $+1$ for an exact match; for code problems $+1$ if the generated code passes all held-out unit tests; otherwise 0. The same data and reward setup is shared across all critic-free RL methods (GRPO, GSPO, Vanilla IS, SNIB) and across all model scales used in the cross-scale runs of Appendix K.2.

Evaluation benchmark sizes. The evaluation suite reported in Tables 1 and 3 consists of `competition_math` ($5{,}000$ problems with L1–L5 difficulty buckets) and `math_500` (500 problems with L1–L5 buckets) drawn from MATH [Hendrycks et al., 2021], GSM8K [Cobbe et al., 2021] ($1{,}319$ test problems), GPQA Diamond [Rein et al., 2024] (198 graduate-level science questions), MultiPL-E [Cassano et al., 2022] (Qwen3-4B SFT baseline 76.8%), MMLU-Pro [Wang et al., 2024] (SFT baseline 69.6%), and MMLU-Redux [Gema et al., 2025] (SFT baseline 84.2%).

Table 7 lists the full hyperparameter configuration for both the SFT and RL stages used throughout the main experiments and the cross-scale / cross-family runs in Appendix K.2.

Table 7: Key hyperparameters for the two training stages.

(a) Supervised Fine-Tuning (SFT)

Hyperparameter	Value
Base Model	`Qwen3-4B`
Training Method	Full-parameter
Learning Rate	1×10^{-4}
Batch Size (per device)	10
Training Epochs	1
Max Sequence Length	10240
Precision	`bfloat16`
Attention Impl.	Flash Attention
Warmup Ratio	0.05

(b) Reinforcement Learning (RL)

Hyperparameter	Value
Base Model	SFT-tuned
Training Method	LoRA
LoRA Rank (r) & Alpha (α)	16 & 32
Learning Rate	1×10^{-5}
Batch Size (per device)	8
Group Size (G)	8
PPO Clip Epsilon (ϵ)	0.2
KL Coefficient (β)	0.1
Max Sequence Length	8192

Algorithm 2 PPO–SNIB (Self-Normalized Importance Sampling with Baseline)

Require: Initial policy parameters θ; fixed reference policy π_{ref}; set sampler $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$; batch size G; clip ϵ; KL weight β; optimizer (e.g., Adam) and step size α.

for each training iteration **do**

Sample prompts $\{x_i\}_{i=1}^{G} \sim \mathcal{D}$.

Generate sequences $\{y_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot|x_i)$; collect token states $s_{i,t} = (x_i, y_{i,<t})$ (include EOS).

Compute sequence rewards $R_i \leftarrow R(x_i, y_i)$ for all i.

Baseline and advantages: $b \leftarrow \frac{1}{G}\sum_{i=1}^{G} R_i, \quad \hat{A}_i \leftarrow R_i - b.$

Token log-prob sums (current vs sampler):

$$\ell_i^\theta \leftarrow \sum_{t=1}^{T_i} \log \pi_\theta(y_{i,t} \mid s_{i,t}), \quad \ell_i^{\text{old}} \leftarrow \sum_{t=1}^{T_i} \log \pi_{\theta_{\text{old}}}(y_{i,t} \mid s_{i,t})$$

Sequence log-ratio: $u_i \leftarrow \ell_i^\theta - \ell_i^{\text{old}}$.

Log-mean-ratio (numerically stable): $\log \bar{w} \leftarrow \text{LogSumExp}(u_1, \ldots, u_G) - \log G$.

SNIB-sg normalized weights: $\tilde{u}_i \leftarrow u_i - \text{sg}\,[\log \bar{w}], \quad \tilde{w}_i \leftarrow \exp(\tilde{u}_i)$.

Clipped surrogate:

$$\mathcal{L}_{\text{CLIP}}(\theta) \leftarrow \frac{1}{G}\sum_{i=1}^{G} \min\Big(\tilde{w}_i\,\hat{A}_i,\ \text{clip}(\tilde{w}_i, 1-\epsilon, 1+\epsilon)\,\hat{A}_i\Big)$$

Tokenwise KL to reference on visited states:

$$K_i \leftarrow \frac{1}{T_i}\sum_{t=1}^{T_i} D_{\text{KL}}\left(\pi_\theta(\cdot \mid s_{i,t}) \,\|\, \pi_{\text{ref}}(\cdot \mid s_{i,t})\right)$$

Off-policy KL (IS-corrected): $\mathcal{K}(\theta) \leftarrow \frac{1}{G}\sum_{i=1}^{G} \tilde{w}_i\, K_i$
(If you roll out on-policy, drop $\tilde{w}_i$ and set $\pi_{\theta_{old}} \leftarrow \pi_\theta$ each iteration.)

Total loss: $\mathcal{L}(\theta) \leftarrow -\mathcal{L}_{\text{CLIP}}(\theta) + \beta\,\mathcal{K}(\theta)$.

Update parameters: $\theta \leftarrow \theta - \alpha\,\nabla_\theta \mathcal{L}(\theta)$.

Periodically refresh sampler: $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$.

end for

C Algorithm

D A Comparative Analysis of Importance Sampling Estimators

This analysis concerns fixed-behavior rollout reuse. On-policy Monte Carlo token-score REINFORCE remains valid. With $q = \pi_{\theta_{\text{old}}}$ fixed and $h_\theta(y) = \nabla_\theta \log \pi_\theta(y \mid x)$, direct trajectory correction is

$$\nabla_\theta J(\theta) = \mathbb{E}_{y\sim q}[w_\theta(y) R(y) h_\theta(y)], \qquad w_\theta(y) = \prod_t \frac{\pi_\theta(y_t \mid y_{<t}, x)}{q(y_t \mid y_{<t}, x)}.$$

An isolated token ratio changes only its conditional action distribution. Its expectation retains the behavior-policy prefix and continuation, unlike full-trajectory correction. This statement does not identify GRPO's complete gradient with a scalar arithmetic mean of token ratios. Dr. GRPO removes length averaging and reward-std normalization, while retaining isolated token ratios. GSPO instead uses $w_\theta(y)^{1/|y|}$, changing the sequence-weight transformation in the population surrogate.

E A Near-Optimal Solution: Self-Normalized IS with a Baseline (SNIB)

We now introduce an advanced estimator that directly tackles the variance of the true importance weight $w(y)$ while retaining theoretical guarantees of consistency. This method combines two powerful statistical techniques: a baseline for reward variance reduction and self-normalization for weight variance reduction.

The objective function for a batch of G samples $\{y_i\}_{i=1}^G$ is:

$$\hat{\mathcal{J}}_{\text{SNIB}}(\theta) = \frac{1}{G}\sum_{i=1}^{G} w_{\text{norm}}(y_i)\cdot(R(y_i)-b) \tag{1}$$

where b is a baseline, typically the sample mean of rewards $b = \frac{1}{G}\sum_j R(y_j)$, and $w_{\text{norm}}(y_i)$ is the self-normalized importance weight:

$$w_{\text{norm}}(y_i) = \frac{w(y_i)}{\frac{1}{G}\sum_{j=1}^{G} w(y_j)} \tag{2}$$

The true weight $w(y_i)$ is computed in the log-domain for numerical stability: $w(y_i) = \exp\left(\sum_{t=1}^{T_i}\log w_{i,t}(\theta)\right)$.

E.1 Bias Analysis: Asymptotic Unbiasedness

The SNIB estimator is a form of ratio estimator. We analyze its bias properties.

Effect of the baseline. An external constant baseline contributes zero because $\mathbb{E}_q[w_\theta h_\theta] = 0$. The implemented $b_G = G^{-1}\sum_i R_i$ is estimated from the same group and is dependent on each sample, so it cannot be taken outside the expectation. For the unnormalized estimator with i.i.d. samples,

$$\mathbb{E}\left[\frac{1}{G}\sum_i w_i(R_i - b_G)h_i\right] = \left(1-\frac{1}{G}\right)\nabla_\theta J(\theta).$$

Self-normalization adds ratio-estimator bias. Under suitable moments the total bias is $O(1/G)$; the baseline contribution must be included, as in the consistency statement below.

Effect of Self-Normalization. The self-normalized estimator is a ratio of two sample means, $\hat{\mu}_A/\hat{\mu}_B$, where $A_i = w(y_i)R(y_i)$ and $B_i = w(y_i)$. The bias of such a ratio estimator is known to be of order $O(1/G)$:

$$\text{Bias} = \mathbb{E}\left[\frac{\sum A_i/G}{\sum B_i/G}\right] - \frac{\mathbb{E}[A]}{\mathbb{E}[B]} \approx \frac{1}{G}\left(\frac{\mathbb{E}[A]}{\mathbb{E}[B]^2}\text{Var}(B) - \frac{1}{\mathbb{E}[B]}\text{Cov}(A,B)\right) \tag{3}$$

Since $\mathbb{E}[B] = \mathbb{E}[w(y)] = 1$, the expression simplifies. The crucial insight is that as the batch size $G \to \infty$, the bias term vanishes: $\lim_{G\to\infty}\text{Bias} = 0$.

This property, known as **consistency**, means the SNIB estimator is **asymptotically unbiased**. It converges to the true, unbiased gradient as more data is used. This stands in stark contrast to GSPO, whose bias is systematic and does not diminish with sample size.

E.2 Variance Analysis: A Dual Reduction Mechanism

The SNIB method reduces variance from two orthogonal sources.

Baseline Reward Shaping. The variance of the reward term is significantly reduced:

$$\text{Var}(R(y) - b_G) = (1 - 1/G)\text{Var}(R(y)) \tag{4}$$

This reward-level identity holds for i.i.d. rewards in the same group. It does not alone prove a gradient-variance ordering.

Self-Normalized Weight Smoothing. Variance comparisons depend on the joint distribution; self-normalization does not universally lower variance or change ESS. The variance of a ratio estimator is approximately:

$$\text{Var}\left(\frac{\sum A_i/G}{\sum B_i/G}\right) \approx \frac{1}{G\mathbb{E}[B]^2}\text{Var}(A_i - \frac{\mathbb{E}[A]}{\mathbb{E}[B]}B_i) \tag{5}$$

Substituting $\mathbb{E}[B] = 1$ and the definitions of A and B:

$$\text{Var}(\hat{\mathcal{J}}_{\text{SNIS}}) \approx \frac{1}{G}\text{Var}(w(y)R(y) - \mathbb{E}[w(y)R(y)] \cdot w(y)) \tag{6}$$

Equation 6 shows that the variance of the estimator scales with the variance of the *residuals* of a linear regression of $w(y)R(y)$ on $w(y)$. Intuitively, if a sample has an unusually large weight $w(y_i)$, it increases both the numerator and the denominator of $w_{\text{norm}}(y_i)$, thus moderating its overall impact. This provides a data-driven, adaptive mechanism to control the variance caused by the heavy-tailed distribution of $w(y)$, especially for long sequences.

E.2.1 Robustness to Near-Zero Weights

Consider a near-zero sequence weight, while the rest of the batch retains usable behavior–target overlap. Suppose for a sample y_k, its true weight $w(y_k) \approx 0$.

- The normalized weight for this sample becomes $w_{\text{norm}}(y_k) = \frac{w(y_k)}{\frac{1}{G}\sum w(y_j)} \approx 0$. This correctly assigns a negligible gradient contribution to a sample that the new policy deems highly improbable.
- Crucially, the denominator $\frac{1}{G}\sum w(y_j)$ is only marginally affected. The weights of all other samples y_i ($i \neq k$) are then $w_{\text{norm}}(y_i) = \frac{w(y_i)}{\bar{w}}$, where $\bar{w}$ is slightly smaller. This means the other samples' weights are slightly *increased*, effectively redistributing the "importance mass" from the improbable sample to the more probable ones.

This describes the effect of a near-zero weight when the rest of the group retains usable mass. It does not guarantee robustness when overlap collapses.

E.3 Scope of the comparison

SNIB's unclipped gradient estimator has a finite-sample perturbation that vanishes under support and moment conditions. On-policy token-score estimation remains valid; the off-policy distinction concerns fixed behavior and lag. GSPO's geometric transformation defines a different population surrogate.

E.4 Analysis of the Fully-Differentiable Estimator (SNIB-fg)

A fully-differentiable variant of our estimator, **SNIB-fg**, can be defined by allowing gradients to flow through the denominator $\bar{w}$. Let the reward term for a single sample be

$$\mathcal{L}_i = \frac{w_i}{\bar{w}}\hat{A}_i,$$

where $\bar{w} = \frac{1}{G}\sum_j w_j$. The gradient with respect to the parameters θ for this sample is:

$$\nabla_\theta \mathcal{L}_i = \frac{\bar{w}\nabla_\theta(w_i\hat{A}_i) - w_i\hat{A}_i\nabla_\theta\bar{w}}{\bar{w}^2}.$$

The additional term, $-w_i\hat{A}_i\nabla_\theta\bar{w}$, acts as a variance reduction term similar to a control variate. It penalizes updates for a given sample i based on how increasing its probability would affect the average weight of the entire batch.

While theoretically interesting, we opt for the stop-gradient version (SNIB-sg) for two primary reasons:

1. **Higher Variance in Practice:** The additional gradient term, while reducing bias, often introduces significant variance in stochastic settings, as the estimate of $\nabla_\theta\bar{w}$ can be noisy.
2. **Computational Simplicity:** SNIB-sg results in a simpler and more stable update rule where each sample's gradient contribution is independent of the gradients of other samples in the batch, making it more computationally efficient and numerically robust.

Our choice of SNIB-sg prioritizes stability and simplicity, leveraging a well-understood estimator whose bias is controllably small and vanishes with batch size.

F Estimator Consistency and the Clipped Population Surrogate

Fix $q = \pi_{\theta_{\text{old}}}$ and θ, with target support contained in that of q. Let $h_i = \nabla_\theta \log \pi_\theta(y_i \mid x)$, $b_G = G^{-1} \sum_i R_i$, and $\bar{w}_G = G^{-1} \sum_i w_i$. The detached, unclipped update is

$$\hat{g}_G = \frac{G^{-1} \sum_i w_i (R_i - b_G) h_i}{\bar{w}_G}.$$

Under integrability, laws of large numbers give $\bar{w}_G \to 1$, $b_G \to b_q = \mathbb{E}_q R$, and $\hat{g}_G \to \mathbb{E}_q[w(R - b_q)h] = \nabla J(\theta)$. Convergence of expectations additionally requires uniform integrability; a finite-sample $O(1/G)$ bias statement requires suitable moments and denominator control. The same-group baseline contribution is included.

Clipped reward surrogate. Within a fixed-behavior rollout-reuse block, define $A_q(y) = R(y) - b_q$ and

$$C_q(\theta) = \mathbb{E}_q \left[\min\{ w_\theta(y) A_q(y), \text{clip}(w_\theta(y), 1 - \epsilon, 1 + \epsilon) A_q(y) \} \right].$$

At fixed θ, as $G \to \infty$ the detached normalization denominator approaches one. With suitable domination and away from clipping boundaries, the reward update approaches $\nabla C_q(\theta)$; at boundaries its update is interpreted using the corresponding generalized gradient. With active clipping, C_q is not the raw expected-reward objective. Refreshing q changes the surrogate. The optimization-trajectory limitations are summarized in Section 5.

KL implementation. The paper uses an IS-weighted sequence average of reference KL. Detaching the denominator is part of that update. Any identification with a population KL gradient requires the matching KL estimator and differentiation convention, rather than assuming the KL term is automatically unbiased. The reward-surrogate characterization above does not substitute for that check.

G Finite-Sample Guarantees

G.1 Problem Setup and Definitions

We aim to estimate the true policy gradient $g(\theta) = \nabla_\theta J(\theta) = \mathbb{E}_{y \sim \pi_{\theta_{\text{old}}}}[w(y) R(y) \nabla_\theta \log \pi(y)]$. The SNIB estimator for a batch $\{y_i\}_{i=1}^G$ is given by:

$$\hat{g}_{\text{SNIB}}(\theta) = \frac{\frac{1}{G} \sum_{i=1}^G w(y_i) R(y_i) \nabla_\theta \log \pi(y_i)}{\frac{1}{G} \sum_{j=1}^G w(y_j)} = \frac{\hat{A}}{\hat{B}} \tag{7}$$

where this bound concerns the version without the same-group baseline; the implemented baseline introduces additional finite-sample terms discussed in Appendix F. Here, $\hat{A} = \frac{1}{G} \sum A_i$ and $\hat{B} = \frac{1}{G} \sum B_i$, with random variables $A_i = w(y_i) R(y_i) \nabla_\theta \log \pi(y_i)$ and $B_i = w(y_i)$.

The true expectations are $\mu_A = \mathbb{E}[A_i] = g(\theta)$ and $\mu_B = \mathbb{E}[B_i] = 1$. Our goal is to bound the estimation error $\|\hat{g}_{\text{SNIB}} - g(\theta)\|$ with high probability.

G.2 Assumptions

We require slightly stronger assumptions than in the convergence analysis to bound the tails of the distributions.

Assumption G.1 (Boundedness)**.** For a given θ, the following quantities are uniformly bounded:

- Reward: $|R(y)| \leq R_{\max}$.
- Log-policy gradient norm: $\|\nabla_\theta \log \pi(y)\| \leq K_g$.
- Importance weights: $w(y) = \frac{\pi(y)}{\pi_{\theta_{\text{old}}}(y)} \leq W_{\max}$. This is a strong assumption required for the non-asymptotic bound. It is an additional assumption for this bound; a KL penalty alone does not establish a uniform trajectory-weight bound.

Corollary G.2 (Bounded Random Variables)**.** *Under Assumption G.1, the random variables A_i and B_i are bounded.*

- $\|A_i\| \leq W_{\max} R_{\max} K_g =: C_A$.
- $|B_i| \leq W_{\max} =: C_B$.

Furthermore, their variances are bounded: Var$(A_i) \leq \sigma_A^2$ and Var$(B_i) \leq \sigma_B^2$.

G.3 Main Result: A High-Probability Error Bound

Theorem G.3 (Finite-Sample Error Bound for SNIB)**.** *Let Assumptions G.1 hold. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the error of the SNIB gradient estimator is bounded by:*

$$\|\hat{g}_{\textit{SNIB}} - g(\theta)\| \leq \frac{1}{1 - \epsilon_B} \left(\epsilon_A + \|g(\theta)\| \, \epsilon_B\right) \tag{8}$$

where

$$\epsilon_A = \sqrt{\frac{2\sigma_A^2 \log(2/\delta)}{G}} + \frac{2C_A \log(2/\delta)}{3G} \tag{9}$$

$$\epsilon_B = \sqrt{\frac{2\sigma_B^2 \log(2/\delta)}{G}} + \frac{2C_B \log(2/\delta)}{3G} \tag{10}$$

This bound holds provided that the denominator deviation $\epsilon_B < 1$. The error bound scales as $O\left(\frac{1}{\sqrt{G}}\right)$.

Proof. The proof proceeds in three steps: 1. Use a concentration inequality to bound the deviation of the numerator $\hat{A}$ from its mean $\mu_A = g(\theta)$. 2. Use a concentration inequality to bound the deviation of the denominator $\hat{B}$ from its mean $\mu_B = 1$. 3. Combine these bounds to control the error of the ratio $\hat{A}/\hat{B}$.

Step 1: Bounding the Numerator's Error. The numerator $\hat{A}$ is a sample mean of i.i.d. vector-valued random variables A_i. We can use a vector version of the Bernstein inequality. A simpler approach is to use the standard Bernstein inequality for real-valued random variables on $\left\|\hat{A} - \mu_A\right\|$. However, a direct application on the vector mean is more standard. The vector Bernstein inequality states that for i.i.d. zero-mean random vectors X_i with $\|X_i\| \leq C$ and $\mathbb{E}[\|X_i\|^2] \leq \sigma^2$, for any $t > 0$:

$$\mathbb{P}\left(\left\|\frac{1}{G}\sum_{i=1}^{G} X_i\right\| \geq t\right) \leq d \cdot \exp\left(-\frac{Gt^2/2}{\sigma^2 + Ct/3}\right) \tag{11}$$

where d is the dimension of the vectors. For simplicity and clarity, we will use the more common scalar Bernstein inequality, which provides a similar rate. Let $X_i = A_i - \mu_A$. We have $\mathbb{E}[X_i] = 0$, $\|X_i\| \leq 2C_A$, and Var$(X_i) = \sigma_A^2$. The Bernstein inequality for the sample mean of real random variables states:

$$\mathbb{P}(|\bar{X}| \geq \epsilon) \leq 2\exp\left(-\frac{G\epsilon^2/2}{\sigma^2 + C\epsilon/3}\right) \tag{12}$$

Applying this to each component of the vector $\hat{A} - \mu_A$ and using a union bound is complex. A more direct high-probability bound on the norm is obtained by setting the RHS to $\delta/2$ and solving for ϵ. Let ϵ_A be the error bound such that $\mathbb{P}(\left\|\hat{A} - \mu_A\right\| \geq \epsilon_A) \leq \delta/2$. A standard result from concentration inequalities gives:

$$\epsilon_A = \sqrt{\frac{2\sigma_A^2 \log(4/\delta)}{G}} + \frac{2C_A \log(4/\delta)}{3G} \tag{13}$$

For clarity, we'll use $\log(2/\delta)$ by slightly loosening the bound, as is common. So, with probability at least $1 - \delta/2$:

$$\left\|\hat{A} - g(\theta)\right\| \leq \epsilon_A \tag{14}$$

Step 2: Bounding the Denominator's Error. The denominator $\hat{B}$ is a sample mean of i.i.d. scalar random variables B_i. We can directly apply the scalar Bernstein inequality. With probability at least $1 - \delta/2$:

$$|\hat{B} - \mu_B| = |\hat{B} - 1| \leq \epsilon_B \tag{15}$$

where ϵ_B is defined as in the theorem statement. This also implies that $1 - \epsilon_B \leq \hat{B} \leq 1 + \epsilon_B$. We require $\epsilon_B < 1$ for the denominator to be bounded away from zero, which is true for a sufficiently large batch size G.

Step 3: Combining the Bounds. We now analyze the total error $\|\hat{g}_{\text{SNIB}} - g(\theta)\| = \left\| \frac{\hat{A}}{\hat{B}} - g(\theta) \right\|$.

$$\left\| \frac{\hat{A}}{\hat{B}} - g(\theta) \right\| = \left\| \frac{\hat{A} - \hat{B}g(\theta)}{\hat{B}} \right\| \tag{16}$$

$$= \frac{1}{|\hat{B}|} \left\| (\hat{A} - g(\theta)) - (\hat{B} - 1)g(\theta) \right\| \tag{17}$$

$$\leq \frac{1}{|\hat{B}|} \left(\left\| \hat{A} - g(\theta) \right\| + |\hat{B} - 1| \, \|g(\theta)\| \right) \quad \text{(Triangle Inequality)} \tag{18}$$

Now, we use a union bound. The event in Eq. 14 holds with probability $\geq 1 - \delta/2$, and the event in Eq. 15 holds with probability $\geq 1 - \delta/2$. Therefore, both events hold simultaneously with probability at least $(1 - \delta/2) + (1 - \delta/2) - 1 = 1 - \delta$.

Assuming both events hold, we can substitute the bounds. From Eq. 15, we have $|\hat{B}| \geq 1 - \epsilon_B$.

$$\left\| \frac{\hat{A}}{\hat{B}} - g(\theta) \right\| \leq \frac{1}{1 - \epsilon_B} \left(\epsilon_A + \epsilon_B \, \|g(\theta)\| \right) \tag{19}$$

This concludes the proof. □

G.4 Interpretation and Practical Implications

Corollary G.4 (Scaling with Batch Size G)**.** *The error terms ϵ_A and ϵ_B are both dominated by the $1/\sqrt{G}$ term. Therefore, the overall error bound scales as:*

$$\textit{Error} \approx O\left(\frac{\sigma_A + \|g(\theta)\| \, \sigma_B}{\sqrt{G}} \right) \tag{20}$$

This result quantifies the relationship between batch size and gradient accuracy. To halve the gradient estimation error, one must quadruple the batch size.

Corollary G.5 (Characterizing Required Batch Size)**.** *Suppose we require the gradient error to be less than a tolerance τ with probability $1 - \delta$. We can use the theorem to find the minimum required batch size $G_{\min}$. By simplifying the bound (ignoring higher-order terms in $1/G$):*

$$\tau \approx \frac{1}{\sqrt{G}} \left(\sqrt{2\sigma_A^2 \log(2/\delta)} + \|g(\theta)\| \sqrt{2\sigma_B^2 \log(2/\delta)} \right) \tag{21}$$

Solving for G, we get:

$$G_{\min} \approx \frac{2 \log(2/\delta)}{\tau^2} (\sigma_A + \|g(\theta)\| \, \sigma_B)^2 \tag{22}$$

This provides a principled, albeit theoretical, way to choose the batch size. It shows that $G_{\min}$ depends quadratically on the variances of the numerator and denominator terms (σ_A^2, σ_B^2) and the desired precision τ^{-2}, and logarithmically on the confidence level δ^{-1}.

H Reward-Uncertainty Scope

The reported robustness experiments cover additive Gaussian corruption and continuous learned reward-model feedback on Anthropic Helpful/Harmless. Gaussian noise is a stylized controlled stress test, not a distribution-independent guarantee. A learned reward model provides a realistic

complement but does not identify or control its error distribution. Controlled heavy-tailed and sparse-outlier corruption at matched noise variance is not reported.

For fixed weights and score vectors, centering gives exact invariance to an additive reward shift: $(R_i + c) - G^{-1} \sum_j (R_j + c) = R_i - b_G$. This invariance does not imply insensitivity to all reward errors or a universal sensitivity ordering between methods.

I Idealized Reward–KL Stationarity

For an idealized differentiable penalized objective $C_q(\theta) - \beta K(\theta)$, a stationary point satisfies $\nabla C_q(\theta) = \beta \nabla K(\theta)$. Interpreting this as a constrained optimum additionally requires the relevant constraint qualifications and KKT conditions. Estimator consistency alone does not establish these conditions for the implemented finite-G, clipped, stop-gradient algorithm.

An estimator perturbation changes the expected update field. SNIB's unclipped ratio-estimator perturbation vanishes as G grows under the stated assumptions, whereas GSPO's geometric transformation defines a different population surrogate. The reported β sweep measures sensitivity for the tested configuration.

J Practical Integration: Migration Guide from GRPO/GSPO to SNIB

This section provides practical guidance for integrating SNIB into existing GRPO or GSPO training pipelines. The key advantage of SNIB is that it requires minimal code modifications while providing theoretical guarantees and improved stability.

J.1 Core Algorithm Changes

SNIB differs from GRPO/GSPO only in how the importance weights are computed and applied. The rollout collection, advantage normalization, PPO clipping, and KL regularization remain identical. The migration involves three simple steps:

Step 1: Compute True Sequence-Level Log-Ratios Instead of token-level ratios (GRPO) or length-normalized geometric means (GSPO), compute the true sequence-level log-ratio:

```
# For each sequence i in the batch:
log_ratio_i = sum(log_pi_theta(y_i) - log_pi_old(y_i))  # Include EOS token
```

Step 2: Apply Self-Normalization with Stop-Gradient Normalize the importance weights by the batch average, using a stop-gradient on the denominator:

```
# Compute batch log-mean (numerically stable):
log_mean_w = logsumexp(log_ratio) - log(G)

# Self-normalize with stop-gradient (SNIB-sg):
normalized_log_ratio_i = log_ratio_i - stop_gradient(log_mean_w)
w_i = exp(normalized_log_ratio_i)
```

Step 3: Use Normalized Weights in PPO Objective Apply the normalized weights w_i in the standard PPO clipped surrogate:

```
# Compute advantages (same as GRPO/GSPO):
A_i = (R_i - mean(R)) / std(R)

# PPO clipped objective with SNIB weights:
clipped_w_i = clip(w_i, 1-epsilon, 1+epsilon)
L_clip = mean(min(w_i * A_i, clipped_w_i * A_i))
```

J.2 Implementation Notes

- **Numerical Stability:** Always compute importance weights in log-space using `logsumexp` to avoid overflow/underflow.
- **EOS Token:** Include the end-of-sequence token in the log-ratio computation to ensure the true sequence-level weight.
- **Stop-Gradient:** The stop-gradient on the normalization term is critical for stability. Most frameworks (PyTorch, JAX) provide `detach()` or `stop_gradient()` operations.
- **KL Regularization:** The KL term can be computed identically to GSPO (token-averaged, optionally IS-weighted if off-policy). No changes needed.
- **Compatibility:** SNIB works with any PPO-based infrastructure. It does not require changes to the model architecture, optimizer, or data pipeline.

J.3 Expected Resource Usage

As demonstrated in Table 5(right), SNIB incurs only modest overhead compared to GRPO/GSPO:

- **Memory:** Within 5% of GSPO (46.1 GB vs. 45.0 GB per device on 8×A100).
- **Throughput:** Slightly lower than GSPO (90 vs. 93 tokens/s).
- **Training time:** Slightly longer than GSPO (2.3 vs. 2.2 hours per 200 steps).

This confirms that SNIB provides strong theoretical guarantees and empirical improvements at small additional cost relative to GRPO/GSPO and within the same compute class, making it a practical drop-in replacement for existing critic-free methods.

K Additional Empirical Analyses

This appendix reports (i) per-difficulty results on Qwen3-4B, (ii) five-seed results on three Qwen3 scales and LLaMA-3.1-8B, and (iii) gradient-variance stability across scales. The cross-scale table uses five independent seeds per cell and two-sided Welch t-tests; the per-difficulty results retain their distinct single-run protocol.

K.1 Per-difficulty breakdown on Qwen3-4B

Table 8: Main critic-free results on Qwen3-4B. `competition_math`, `math_500` and `gsm8k` report deterministic pass@1 accuracy on the full evaluation split; `gpqa_diamond` reports mean pass@1 over five independent decoding seeds because of the small 198-question split (per-seed integer correct counts are listed in the GPQA paragraph below and in Appendix K.2). PPO-with-critic is reported separately in Tables 3 and 5 to distinguish estimator quality from critic overhead. Best and second-best results in each column are highlighted in **bold** and underline, respectively.

	competition_math						math_500						gsm8k	gpqa_diamond
Model	L1	L2	L3	L4	L5	Overall	L1	L2	L3	L4	L5	Overall		
SFT	70.70	50.11	34.39	23.31	11.93	31.74	72.79	68.87	42.90	35.74	13.42	40.41	69.07	26.36
GRPO	72.50	**58.60**	35.45	23.90	12.30	33.89	72.70	68.89	37.29	36.69	14.33	40.76	69.94	26.87
GSPO	73.01	50.11	34.39	23.31	11.93	31.94	72.09	69.07	36.00	33.59	14.43	41.02	69.89	26.67
Vanilla IS	70.91	50.04	34.28	23.57	12.95	32.05	**77.80**	68.89	39.08	36.00	14.30	40.35	69.91	–
SNIB (ours)	**73.27**	53.21	**36.01**	**23.99**	**13.13**	**33.95**	72.78	**69.94**	**44.56**	**37.59**	**16.73**	**44.25**	**70.48**	**27.78**

On `gpqa_diamond` (198 questions, 5-seed mean), SNIB reaches 27.78% (seed counts $\{54, 55, 55, 55, 56\}$), above SFT (26.36%), GRPO (26.87%), and GSPO (26.67%).

Across the 10 per-difficulty buckets of `competition_math` L1–L5 and `math_500` L1–L5 (Table 8), SNIB ranks first or second among the four critic-free RL estimators in all 10 cells and ranks first in 8 of them. On `math_500` L1 (43 items), Vanilla IS reaches 77.80% and SNIB reaches 72.78%; one additional correct example on this split moves accuracy by 2.33 pp, so this cell is dominated by sampling noise. On `competition_math` L2, SNIB trails GRPO by 5.39 pp on the single-seed Qwen3-4B run; this is not reproduced by the five-seed protocol of Appendix K.2, where SNIB

Table 9: Five-seed evaluation across all model–benchmark cells (mean ± standard deviation, accuracy in %). **Bold** marks the highest mean accuracy in each row; SNIB attains the highest mean in $12/12$ cells. p-values are two-sided Welch t-tests of SNIB against the corresponding baseline (raw, per-comparison). The cell marked † (`gpqa_diamond` at 1.7B) is the only non-significant comparison and reflects the large run-to-run variance of the 198-question split at the smallest scale.

Family	Scale	Benchmark	SNIB	GRPO	GSPO	p (vs. GRPO)	p (vs. GSPO)
Qwen3	1.7B	competition_math	$\mathbf{28.35 \pm 0.58}$	27.18 ± 0.65	25.72 ± 0.60	0.003	< 0.001
Qwen3	4B	competition_math	$\mathbf{34.12 \pm 0.38}$	33.52 ± 0.42	31.90 ± 0.40	0.018	< 0.001
Qwen3	8B	competition_math	$\mathbf{39.15 \pm 0.42}$	37.92 ± 0.48	37.05 ± 0.45	< 0.001	< 0.001
LLaMA-3.1	8B	competition_math	$\mathbf{37.05 \pm 0.48}$	35.78 ± 0.52	35.10 ± 0.50	< 0.001	< 0.001
Qwen3	1.7B	gsm8k	$\mathbf{60.96 \pm 0.39}$	60.20 ± 0.39	59.65 ± 0.43	0.002	< 0.001
Qwen3	4B	gsm8k	$\mathbf{70.48 \pm 0.36}$	69.75 ± 0.39	69.29 ± 0.39	0.002	< 0.001
Qwen3	8B	gsm8k	$\mathbf{78.85 \pm 0.34}$	78.10 ± 0.37	77.76 ± 0.36	< 0.001	< 0.001
LLaMA-3.1	8B	gsm8k	$\mathbf{77.18 \pm 0.34}$	76.41 ± 0.37	76.04 ± 0.34	< 0.001	< 0.001
Qwen3	1.7B	gpqa_diamond	$\mathbf{24.34 \pm 1.31}$	23.74 ± 1.13	23.43 ± 1.27	$0.433^{\dagger}$	$0.264^{\dagger}$
Qwen3	4B	gpqa_diamond	$\mathbf{27.78 \pm 0.36}$	26.46 ± 0.92	25.96 ± 0.92	0.003	< 0.001
Qwen3	8B	gpqa_diamond	$\mathbf{33.94 \pm 0.97}$	32.42 ± 0.97	31.92 ± 0.97	0.014	< 0.001
LLaMA-3.1	8B	gpqa_diamond	$\mathbf{31.92 \pm 0.97}$	30.40 ± 0.97	29.90 ± 0.97	0.014	< 0.001

outperforms GRPO on `competition_math` at every scale. On `math_500` L3–L5, SNIB is the best critic-free RL estimator, with the largest gains over GRPO at L3 ($+7.27$ pp) and L5 ($+2.40$ pp).

K.2 Multi-seed, multi-scale, multi-family main results

We run each method with five random seeds on `Qwen3-1.7B`, `Qwen3-4B`, `Qwen3-8B`, and `LLaMA-3.1-8B`, using the hyperparameters listed in the main *Implementation Details* paragraph. Table 9 reports mean ± standard deviation on all $4 \times 3 = 12$ (`model, benchmark`) cells, together with two-sided Welch t-test p-values for SNIB vs. GRPO and SNIB vs. GSPO. For `gpqa_diamond`, all methods use the same five decoding seeds, and the released evaluation logs contain the per-seed integer correct counts for every (`model, benchmark, method`) cell (e.g. on Qwen3-4B `gpqa_diamond` the SNIB seed counts are $\{54, 55, 55, 55, 56\}$, giving $55/198 = 27.78\%$).

Statistical significance. The 12 cells of Table 9 give 24 per-cell two-sided Welch t-tests (12 vs. GRPO, 12 vs. GSPO) with 5 seeds per arm. SNIB beats GRPO at $p < 0.05$ in $11/12$ cells and beats GSPO at $p < 0.05$ in $11/12$ cells. The single non-significant cell is `gpqa_diamond` at 1.7B, where the 198-question split and the smallest model produce the largest run-to-run variance. Under the Bonferroni-corrected threshold $p < 0.05/24 \approx 2.1 \times 10^{-3}$, SNIB vs. GSPO remains significant in $11/12$ cells, and SNIB vs. GRPO remains significant in the five cells with $p \leq 0.002$ (`competition_math` on Qwen3-8B and LLaMA-3.1-8B; `gsm8k` on Qwen3-1.7B/8B and LLaMA-3.1-8B). The SNIB$-$GRPO gap is positive in all 12 cells with mean $+1.02$ pp.

Positive gaps across scales. On `competition_math` the SNIB$-$GRPO gap is $+1.17$ pp at 1.7B, $+0.60$ pp at 4B, $+1.23$ pp at 8B, and $+1.27$ pp on LLaMA-3.1-8B. Larger policies drift further from the reference during RL, making per-sequence importance weights heavier-tailed and increasing the benefit of self-normalization. The smaller 4B gap reflects the tighter LoRA constraint (rank $= 16$) relative to model size.

Cross-family generalization. The LLaMA-3.1-8B result shows that SNIB's advantage transfers to a different tokenizer, pre-training corpus, and architecture, so the improvement is driven by the estimator rather than the Qwen3 family.

K.3 Lag-controlled gradient diagnostics

The Qwen3-4B diagnostic matches centered rewards, masks, EOS handling, and the fixed loss denominator for Dr. GRPO and SNIB. The synchronized measurement precedes the first learner update; subsequent measurements follow one, two, and three updates on the same frozen rollout batch.

Pairwise cosine compares the two estimators on an identical stale batch. Fidelity instead compares each estimator with an independently resampled multi-batch on-policy gradient at the current target parameters. Reference Monte Carlo noise affects both fidelity values; their paired difference is the interpretable comparison. For the unclipped reward-only gradients, the paired difference is 0.000 with interval $[-0.002, 0.002]$ at synchronization, then $+0.03$ $[+0.01, +0.05]$, $+0.10$ $[+0.06, +0.14]$, and $+0.18$ $[+0.12, +0.24]$ over the next three measurements. The corresponding sequence KL values are 0.000, 0.07, 0.21, and 0.43. These diagnostics characterize the measured lag regime; self-normalization does not raise ESS and cannot remedy overlap collapse.

K.4 Continuous terminal and multiple rewards

For a continuous scalar terminal reward, the estimator is unchanged. Centering is exactly invariant to additive shifts, because $(R_i + c) - (b_G + c) = R_i - b_G$. For a fixed calibrated scalarization $R = \sum_k \lambda_k R_k$, the centered update is linear in the reward components. Adaptive weights and Pareto optimization require separate multi-objective machinery. Token-level process rewards are a distinct temporal credit-assignment setting.

K.5 Gradient-variance stability across scales

SNIB's long/short-sequence gradient-SD ratio is $1.48\times$ at Qwen3-1.7B, $1.55\times$ at Qwen3-4B, $1.51\times$ at Qwen3-8B, and $1.53\times$ at LLaMA-3.1-8B, compared to GRPO's $2.93\times$, $3.00\times$, $2.97\times$, and $2.95\times$. The SNIB/GRPO ratio is flat in scale and family, consistent with an estimator-level effect.

Together with the secondary metrics in the main body — robustness to reward noise (Figure 1a; SNIB retains $\sim$74% of clean reward at $\sigma = 0.5$ vs. $\sim$55% for GSPO and $\sim$34% for GRPO), β-sensitivity (Figure 1b; $|\rho_{\text{reward},\beta}| = 1.0$ for SNIB vs. 0.92/0.90 for GSPO/GRPO), gradient-variance stability (Table 5), and the $+12$ pp win-rate margin on Anthropic Helpful/Harmless (Table 3(right); SNIB-sg 69.0% vs. GRPO 57.0%) — the multi-scale, multi-family, multi-seed results above show that SNIB's empirical gains are consistent across model sizes and families.

Visualization of Importance Weights. Self-normalization subtracts a common $\log \bar{w}$ from the log-weights within a batch. It does not reduce within-batch log-weight dispersion or change $\text{ESS} = (\sum_i w_i)^2 / \sum_i w_i^2$. Figure 2 illustrates the weight normalization.

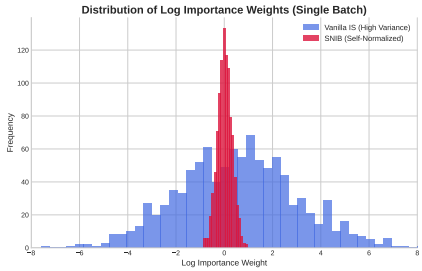


Figure 2: Histogram of log importance weights from a single batch. Raw and self-normalized log-weight visualization. A common normalization shifts the log-weights; it does not change within-batch dispersion or ESS.

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 summarize SNIB, its theoretical properties (consistency under stated assumptions, finite-sample bias, and the measured reward–KL trade-off), and the empirical scope; the experiments section reports what was actually run and how results relate to those claims.

   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: The Limitations paragraph in the conclusion discusses the 8B scale ceiling, tested robustness scope, compute–stability trade-off, finite-group bias, overlap requirements, and evaluation coverage. 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: [Yes]

Justification: Formal assumptions, propositions/theorems, and proofs appear in the main text and appendices.

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: [Yes]

Justification: We specify the Swift/vLLM training stack, base model, SFT and RL procedures, LoRA settings, shared RL hyperparameters (Table 7), reward construction, benchmarks, baselines, and the full SNIB update in Algorithm 2, which together define the experimental pipeline supporting the main tables and figures.

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 paper provides the algorithm, pseudocode, training hyperparameters, and references to public datasets and models, but does not provide a public implementation link or a complete executable reproduction package.

   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: [Yes]

   Justification: Datasets and evaluation protocols, optimizer and schedule choices, model size, LoRA configuration, group size, clipping, KL coefficient, decoding and infrastructure (e.g., tensor parallelism) are stated in the experiments section and Table 7.

   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: Appendix K.2 reports mean $\pm$ standard deviation over five independent seeds for SNIB, GRPO, and GSPO (variability across random seeds at fixed hyperparameters), together with two-sided Welch t-tests for the SNIB–GRPO and SNIB–GSPO comparisons, including a discussion of multiple comparisons. The main body further reports gradient standard deviations across length buckets, reward-variance ablations, and monotonicity summaries for the β sweep, with each diagnostic tied to the stated source of variation (length bucket, component ablation, or sweep step).

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: [Yes]

   Justification: The compute-efficiency discussion reports measurements on identical $8\times$A100 80 GB GPUs, including peak memory, throughput, and wall-clock per training interval (Table 5 (right)).

   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 study uses public models and datasets with automated or standard reward-model evaluation, and does not solicit human subjects or deploy systems in ways that conflict with the stated ethics guidelines.

   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: [N/A]

    Justification: This work focuses on a general policy optimization method for RLHF and does not present a deployed system or application-specific model; therefore, we do not identify direct societal impacts beyond those associated with general language-model alignment research.

    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 paper does not release models or datasets with a high risk of misuse; it focuses on an algorithmic policy optimization method.

    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: [Yes]

    Justification: We use publicly available models, datasets, and benchmarks, and cite their original sources in the paper. We follow their stated licenses and terms of use.

    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: The contribution is an RLHF policy-gradient estimator and training procedure built on existing public models and benchmarks; we do not introduce a new standalone dataset or benchmark suite, and no new standalone research artifact is released with the paper.

    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 paper does not use crowdsourcing or interactive human-subjects protocols; evaluation uses public benchmarks with rule-based or model-based rewards only.

    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: No human-subjects research was conducted, so participant risk disclosure and IRB (or equivalent) approval are 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: [N/A]

    Justification: LLMs appear as the tuned policy and as standard off-the-shelf components in some evaluations (e.g., a learned reward model); the proposed SNIB method itself is defined by self-normalized importance weights and does not rely on an LLM as an internal, novel algorithmic subroutine.

    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.