RENIO: REWEIGHTING NEGATIVE TRAJECTORY IMPORTANCE FOR LLM ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) improves LLM reasoning by training a student model on its own generated outputs, but standard OPD treats all student-generated outputs (SGOs) equally regardless of their informativeness. We observe a consistent asymmetry in controlled filtering experiments: in both OPD and on-policy self distillation (OPSD), training only on incorrect SGOs outperforms training only on correct ones. Our further analysis suggests that models trained on correct-only SGOs tend to generate shorter reasoning traces and show weaker reflection behavior, while incorrect SGOs better preserve exploratory reasoning near the model's capability boundary. To exploit this signal without requiring full answer-containing rollouts, we introduce ReNIO, which **Re**weights **N**egative trajectory **I**mportance for LLM **O**n-policy distillation. By using the student-to-teacher probability ratio, ReNIO identifies pivotal tokens leading to wrong reasoning traces and aggregates their information into a normalized trajectory weight, inherently assigning larger weights to likely negative trajectories without observing the correctness of final-answer. Since ReNIO only uses prefix-conditioned token probabilities, it preserves OPD's prefix training advantage over full-rollout reinforcement learning. Across both mathematical reasoning and code generation tasks, ReNIO improves both OPD and OPSD, with representative relative gains of up to 8.90% for Qwen3-1.7B and 10.00% for R1-Distill-Qwen-7B on mathematical reasoning benchmarks.Code is avaliable at: https://anonymous.4open.science/r/ReNIO-22CDAHGKJH612.

1 INTRODUCTION

Reinforcement learning (RL) and On-Policy distillation (OPD) are becoming two mainstream paradigms for post-training large language models (LLMs) on reasoning tasks (Ke et al., 2026; Chen et al., 2026). RL methods sample on-policy reasoning trajectories and update the model according to externally returned rewards Yu et al. (2025); Bai et al. (2025); Zhihong Shao (2024); DeepSeek-AI et al. (2025). OPD, in contrast, trains on student-generated outputs (SGOs) by aligning the student's token-level distributions with teacher distributions along the same sampled trajectories Lu & Lab (2025); Agarwal et al. (2023). This gives OPD a more dense supervision signal: while RL typically receives sparse sequence-level rewards and must address credit assignment over long trajectories, OPD can provide token-wise feedback at every visited prefix.

OPD also has a substantial training-cost advantage because it can learn from short SGO prefixes instead of full answer-containing rollouts with just little performance degradation Zhang et al. (2026); Zhao et al. (2026). In comparison, reinforcement learning for reasoning tasks must observe the final answer of an on-policy reasoning trajectory before assigning a reward Wen et al. (2025). For difficult reasoning problems, obtaining this answer may require generating very long trajectories, which makes RL substantially more expensive in generation, memory, and training time.

These two advantages make OPD attractive for efficient reasoning post-training, but they also shift the bottleneck from obtaining supervision to deciding which student-generated prefixes deserve more training emphasis. Standard OPD and on-policy self distillation(OPSD)Zhao et al. (2026) assign equal training weight to all SGOs, even though on-policy trajectories can differ greatly in how much useful learning signal they contain. Our controlled filtering experiments reveal a counterintuitive but consistent asymmetry: under both OPD and on-policy self-distillation (OPSD) Zhao et al.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(2026), training only on incorrect SGOs consistently outperforms training only on correct SGOs. Further behavioral analysis shows that correct-only training tends to produce shorter responses with fewer reflection-style markers, whereas incorrect-only training better preserves exploratory and self-corrective reasoning. These findings suggest that negative trajectories can expose valuable process-level correction signals for the student's own failure modes, and may therefore play a more important role in OPD.

Hence, it's important to emphasize incorrect trajectories during OPD. The challenge is to emphasize informative negative trajectories without observing whether the final answer is correct. A direct correctness-based weighting rule would require rolling out each SGO until the answer appears, which would remove the short-prefix efficiency that makes OPD attractive. Moreover, the useful signal in an incorrect SGO is usually localized: a trajectory often becomes wrong because a few pivotal tokens steer the reasoning toward a wrong branch, while many surrounding tokens remain routine continuations. Therefore, an effective weighting signal should be prefix-computable.

We propose ReNIO, a trajectory-level reweighting method that emphasizes negative trajectories using only prefix-conditioned teacher and student probabilities. ReNIO identifies pivotal tokens by the student-to-teacher probability ratio: when the student assigns high probability to a token that the teacher assigns low probability, this ratio serves as prefix-computable evidence for a student-preferred but teacher-rejected branching decision. ReNIO selects such high-ratio tokens with a fixed threshold, aggregates their clipped log ratios by a geometric mean, and normalizes the resulting weights within each batch. The resulting objective redistributes training emphasis toward SGOs that contain strong corrective signals while preserving OPD's ability to train on truncated prefixes.

Our contributions are summarized as follows:

- We identify a negative-trajectory asymmetry in OPD and OPSD: incorrect SGOs can be more useful than correct SGOs for improving reasoning performance.
- We introduce ReNIO, a prefix-computable trajectory reweighting method that uses student-to-teacher log ratios to emphasize likely negative, high-information trajectories without final-answer labels, thereby preserving the short-prefix training advantage of on-policy distillation without additional rollout or reward-labeling cost.
- We show that ReNIO improves both OPD and OPSD across mathematical reasoning and code generation tasks, with representative relative gains of up to 8.90% for Qwen3-1.7B and 10.00% for R1-Distill-Qwen-7B on mathematical reasoning benchmarks.

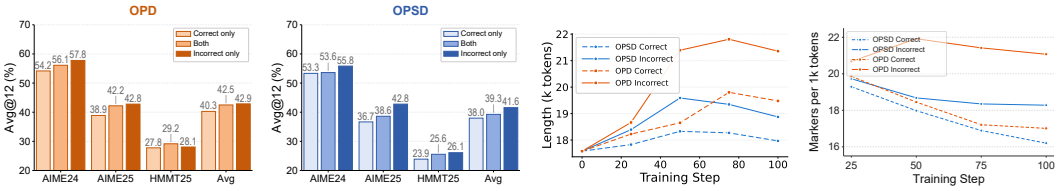
2 Preliminary Analysis of On-Policy Distillation

A natural expectation is that correct SGOs should be the most useful training examples, since they contain successful reasoning paths from the student's own policy. Motivated by findings in reinforcement learning Zhu et al. (2025), we test this assumption on Qwen3-1.7B Yang et al. (2025) by training OPD and OPSD variants using only correct or only incorrect SGOs. Additional setup details are provided in Appendix B.

Figure 1a 1b shows that incorrect-only training consistently outperforms correct-only training under both OPD and OPSD, which means correct and incorrect SGOs are not interchangeable, they may have different roles during training.To better understand this gap, we examine average response length and reflection-style markers per 1k tokens (Figures 1c and 1d).

Incorrect-only training produces longer responses, suggesting that learning from unsuccessful attempts may preserve more extended exploration, whereas correct-only training may favor shorter, familiar solution paths.Length alone, however, does not establish reflective reasoning. After length normalization, marker frequency in Figure 1d declines markedly under correct-only training but is better preserved under incorrect-only training. Here OPSD exhibits a gradual decline even with incorrect SGOs, potentially because standard OPSD makes student much confidence(Kim et al. (2026)). Nevertheless, incorrect-only OPSD retains a higher marker frequency than correct-only OPSD, suggesting better preservation of reflection-style behavior.

These results suggest that correct-only training mainly consolidates behavior the student already performs successfully, but may also **make the model overly confident and less exploratory.** Incorrect-



(a) Correct-only vs. incorrect-only training under OPD.

(b) Correct-only vs. incorrect-only training under OPSD.

(c) Average response length on AIME24.

(d) Reflection-style token count per 1K tokens.

Figure 1: Motivation analysis for correct-only versus incorrect-only on-policy distillation. Left two panels: Avg@12 (%) under OPD and OPSD, where incorrect-only training consistently outperforms correct-only training across mathematical reasoning benchmarks. Right two panels: inference-time behavior on AIME24, where incorrect-only training produces longer responses and more reflection-style markers.

only training exposes failed but structured attempts from the current policy, helping **preserve cautious reasoning behavior**. This interpretation is consistent with prior links between reasoning length, reflection markers, and robustness Kim et al. (2026).

Standard OPD and OPSD nevertheless weight all trajectories equally. Our analysis suggests that trajectories exposing stronger student–teacher deviations deserve more emphasis, directly motivating ReNIO's trajectory-level weighting strategy.

3 Methodology

Figure 2a summarizes ReNIO. We first formalize the on-policy distillation setup (Section 3.1), then introduce a prefix-computable student–teacher log-ratio signal for estimating SGO quality (Section 3.2), finally use a fixed-threshold to select pivotal tokens and aggregate their quality information into a normalized trajectory weight(Section 3.3).

3.1 Problem Setup

Consider a student policy π_S and a teacher policy π_T. Given a prompt x sampled from the training prompt distribution D, on-policy distillation first samples a student-generated output (SGO) $y = (y_1, \ldots, y_n) \sim \pi_S(\cdot|x)$, and then asks the teacher to provide token-level supervision on the same student-visited prefixes. For a model $M \in \{S, T\}$, we write

$$p_M^t(v) = \pi_M(v|x, y_{<t}) \tag{1}$$

for the next-token distribution at position t. We also use $\pi_M(y|x)$ as shorthand for the collection of prefix-conditioned distributions $\{p_M^t\}_{t=1}^n$ along the sampled SGO, rather than only the probability of the completed sequence.

Standard OPD trains the student by matching its prefix-conditioned distributions to the teacher's distributions on every sampled SGO. We define the sequence-level distillation divergence along y as the sum of token-level divergences over all visited prefixes:

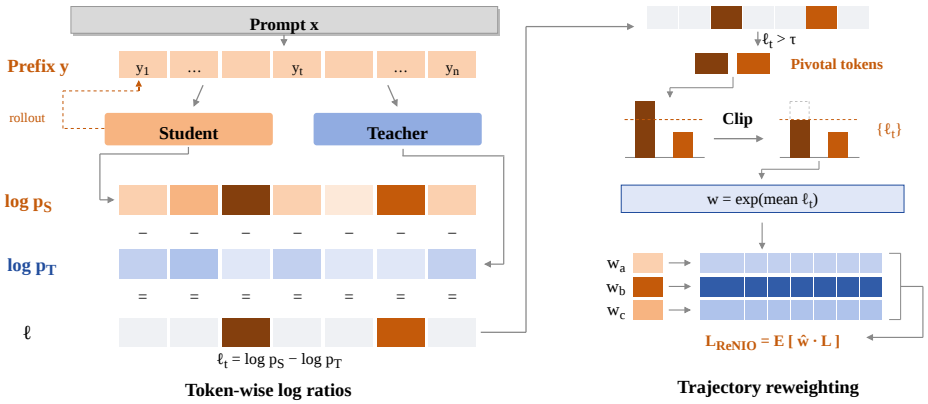
$$\mathcal{D}(\pi_S(y|x)\|\pi_T(y|x)) = \sum_{t=1}^{n} d(p_S^t, p_T^t), \tag{2}$$

where $d(\cdot,\cdot)$ is the distribution divergence, like FKLD, RKLD. Thus, $\mathcal{D}$ does not denote a single divergence between two completed-sequence probabilities; it denotes the accumulated teacher–student discrepancy along the sampled SGO prefixes. The unweighted on-policy objective is therefore

$$L_{\text{OPD}} = \mathbb{E}_{x\sim D,\, y\sim\pi_S(\cdot|x)}\big[\mathcal{D}(\pi_S(y|x)\|\pi_T(y|x))\big]. \tag{3}$$

3.2 Finding an Information Signal for SGO Quality

The central challenge is to identify high-value SGOs without using their final answers, since directly weighting trajectories by correctness would require complete answer-containing rollouts. ReNIO



(a) Overall pipeline of ReNIO.

ANS	…	She	has	9	*	2	=	<<	9	*	2	=	18	>>	…
Teacher		0.67	0.76	**0.21**	**0.00**	1.00	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SGO	…	She	has	**13**	*	2	=	<<	13	*	2	=	26	>>	…
Student		0.74	0.36	**0.78**	**0.13**	0.98	0.96	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Pivotal tokens

(b) An example about how the pivotal tokens leading to a wrong trajectory, and teacher/student probilities on these tokens.

Figure 2: Overview of the proposed ReNIO.Phase I computes the student–teacher log ratio along the SGO; Phase II selects pivotal tokens with a fixed threshold; Phase III aggregates selected log ratios into a normalized trajectory-level weight.

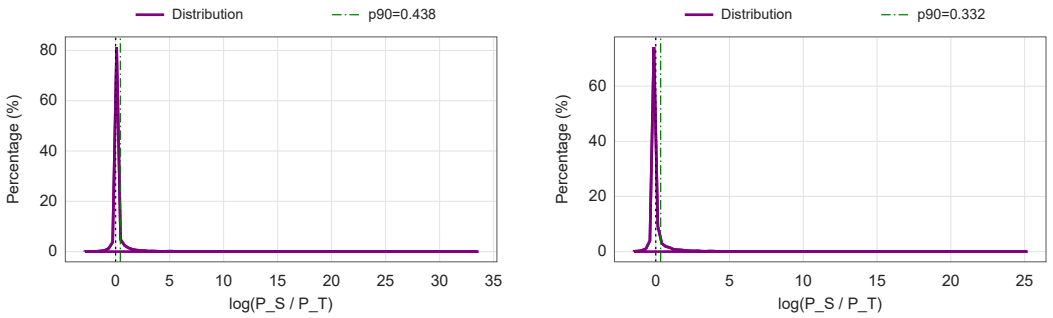
therefore looks for a prefix-computable signal that indicates when a student trajectory is likely to contain an informative failure.

The signal should focus on branching decisions rather than all tokens. A reasoning trajectory usually becomes wrong after a small number of local choices redirect an otherwise plausible solution path. In Figure 2b, for example, the student chooses “13” at a critical step. Because this token is sampled from the student policy, it has relatively high probability under $\pi_S(\cdot|x, y_{<t})$; because it moves the solution away from the teacher-preferred path, it has low probability under $\pi_T(\cdot|x, y_{<t})$. Such tokens therefore have a large student-to-teacher probability ratio. After the wrong branch is taken, later tokens are conditioned on the same mistaken prefix, so teacher–student disagreement may shrink again. We therefore use the student-to-teacher ratio to locate the local decisions where the student is confident and the teacher rejects the sampled choice.

We conduct a toy experiment on GSM8k datasetCobbe et al. (2021) to show how student-to-teacher ratio reflects trajectory-level quality. We select 200 samples from GSM8k, and let student generate corresponding inference trajectories, then identify the first token at which the reasoning becomes erroneous, and calculate their mean student-to-teacher log ratio.

Table 1: Mean student–teacher log ratios on GSM8K. SGO rows use macro averages over per-trajectory token means; the first-error row averages annotated tokens.

Group	Mean log ratio
Correct SGOs	0.019
Incorrect SGOs	0.283
First-error tokens	**4.211**



(a) OPD (b) OPSD

Figure 3: Distribution of token-level log ratios ℓ_t on SGOs in the Qwen3 mathematical reasoning setting. Both OPD and OPSD exhibit long-tailed distributions: most tokens have small log ratios, while a small number of tokens exhibit large student–teacher disagreement.

Table 1 provides two supporting observations. First, teacher-identified first-error tokens have a mean log ratio of 4.211, substantially larger than the trajectory-level averages, showing that error-causing decisions are associated with pronounced log ratio. Second, incorrect SGOs have a higher macro-average log ratio than correct SGOs. This trajectory-level gap is consistent with incorrect SGOs containing more error-related, high-ratio decisions. Together, these results support using token-level log ratios as a prefix-computable signal of SGO quality.

Token-level importance score. For each sampled token y_t in an SGO, we define the student-to-teacher ratio and its log form

$$r_t = \frac{\pi_S(y_t \mid x, y_{<t})}{\pi_T(y_t \mid x, y_{<t})}, \ \ell_t = \log \pi_S(y_t|x, y_{<t}) - \log \pi_T(y_t|x, y_{<t}). \tag{4}$$

A large r_t means that the sampled token is much more likely under the student than under the teacher. We treat high-ratio tokens as ***pivotal tokens***: local decisions where the SGO departs from the teacher-preferred reasoning direction.

The log ratio also has an optimization interpretation. Under a prefix-level reverse-KL objective $\mathrm{KL}(p_S^t \| p_T^t)$, the effective gradient weight of a student-sampled token is $\log \pi_S(y_t|x, y_{<t}) - \log \pi_T(y_t|x, y_{<t}) = \ell_t$ up to a removable baseline. Thus, ℓ_t is both a disagreement signal and a token-level corrective weight.

3.3 From Key Tokens to Trajectory Weight

Figure 3 shows the empirical distribution of ℓ_t in the Qwen3 mathematical reasoning setting. Under both OPD and OPSD, the distribution is strongly long-tailed: most tokens have ℓ_t really close to zero, which means student and teacher align well on most tokens in SGO, while only a small subset has much larger values. This pattern matches the intuition from Section 3.2: **key tokens are really rare, but when they appear, the student-to-teacher ratio is large**.

This long-tailed structure motivates a simple fixed-threshold rule. ReNIO keeps a token as key token if its log ratio exceeds τ, thus result in a key token set:

$$\mathcal{K}(x, y) = \{\, t \mid \ell_t > \tau \,\}, \tag{5}$$

After selecting $\mathcal{K}(x, y)$ for each SGO, we use a **geometric mean** to aggregate its token-level information, equivalently averaging log student-teacher ratios and then exponentiating, resulting the SGO weight. Noticing the log ratio cound have a very large value, we apply a clip operation before exponentiating.

$$w(x, y) = \exp\Big(\frac{1}{L_{\mathcal{K}}} \sum_{t \in \mathcal{K}(x,y)} clip(\log r_t, \epsilon_{max})\Big) \approx \left(\prod_{t \in \mathcal{K}(x,y)} \frac{\pi_S(y_t|x, y_{<t})}{\pi_T(y_t|x, y_{<t})} \right)^{1/L_{\mathcal{K}}} \tag{6}$$

Where $L_{\mathcal{K}} = |\mathcal{K}(a,b)|$, ϵ_{max} is the clip range. When $\mathcal{K}(x,y) = \emptyset$, we will set $w(x,y) = 1$.

To keep the overall gradient scale unchanged, we finally normalize weights within each batch:

$$\hat{w}(x,y) = \frac{w(x,y)}{\bar{w}_B}, \quad \bar{w}_B = \frac{1}{B}\sum_{i=1}^{B} w(x_i,y_i), \tag{7}$$

where B is the batch size, τ is the clip range. The mean weight in each batch is therefore 1, so ReNIO redistributes emphasis across SGOs without changing the average update scale.

Finally we apply trajectory weights calculated by ReNIO to Equation 3 with stop-gradient. The final objective is

$$L_{\text{ReNIO}} = \mathbb{E}_{x\sim D, y\sim\pi_S(\cdot|x)}[\text{sg}(\hat{w}(x,y))\mathcal{D}(\pi_S(y|x)||\pi_T(y|x))] \tag{8}$$

4 Experiments

4.1 Experimental Setup

Models and distillation settings. We evaluate ReNIO on two model families: Qwen3 Yang et al. (2025) and DeepSeek-R1-Distill-Qwen DeepSeek-AI (2025). We study both teacher-based OPD and teacher-free OPSD. For OPD, we use Qwen3-1.7B and DeepSeek-R1-Distill-Qwen-1.5B as students, with Qwen3-8B and DeepSeek-R1-Distill-Qwen-7B as the corresponding teachers. For OPSD, we evaluate Qwen3-1.7B, Qwen3-4B, Qwen3-8B, DeepSeek-R1-Distill-Qwen-1.5B, and DeepSeek-R1-Distill-Qwen-7B.

Tasks and training data. We evaluate ReNIO on two task domains: mathematical reasoning and code generation. For mathematical reasoning, we follow the data construction protocol of OPSD Zhao et al. (2026) and use the same math training data for fair comparison. For code generation, we sample 30k coding-related examples from OpenThoughts Guha et al. (2025).

Evaluation. We evaluate mathematical reasoning on AIME2024 Zhang & Math-AI (2024), AIME2025 Zhang & Math-AI (2025), and HMMT2025 Dekoninck et al. (2026), reporting the average pass@12 (Avg@12) across the three benchmarks. Using evalplusLiu et al. (2023), we evaluate code generation on HumanEval+ Liu et al. (2023) and MBPP+ Liu et al. (2023), reporting average pass@4 (Avg@4).

Implementation details. Following previous workZhao et al. (2026), for OPSD, we fix the teacher parameters to the initial model parameters and use full-vocabulary logit distillation and LoRA for all experiments. We set the maximum generation length to 1024 for Qwen3 OPD and OPSD training and 2048 for DeepSeek-R1-Distill-Qwen OPD and OPSD training; thus, training uses only SGO prefixes. Detailed information is provided in Appendix B.

4.2 Main Results

Tables 2 and 3 report the main results on mathematical reasoning and code generation. ReNIO consistently improves the corresponding OPD or OPSD baseline across all model blocks in terms of average performance: for OPSD, DS-R1-Qwen-7B obtains a 10.00% relative improvement on AIME25; for OPD, DS-R1-Qwen-1.5B obtains a 15.44% relative improvement on HMMT25, showing that reweighting negative on-policy trajectories is beneficial for both teacher-free self-distillation and teacher-based distillation.

Meanwhile, for DS-R1-Qwen-1.5B, standard OPSD slightly decreases the math average compared with the base model, while OPSD+ReNIO improves over both OPSD and the base model on all three math benchmarks. This suggests that ReNIO not only improves average performance, but also stabilizes on-policy self-distillation when the unweighted OPSD signal is noisy.

4.3 Effective and Efficient Training from Short SGO Prefixes

The ability to train on short SGO prefixes is a central practical advantage of OPD and OPSD. Unlike GRPO, which must generate a complete trajectory before assigning a reward, prefix-based distil-

Model	Math Tasks				Coding Tasks		
	AIME24	AIME25	HMMT25	Avg	HumanEval+	MBPP+	Avg
Qwen3-1.7B	51.33	36.00	23.33	36.89	73.63	57.21	65.42
w/ GRPO	49.44	38.61	23.61	37.22	74.84	**65.87**	70.36
w/ OPSD	53.06	41.94	27.50	40.83	74.85	62.43	68.64
w/ OPSD+ReNIO	**57.78** +8.90%	**42.78** +2.00%	**27.78** +1.02%	**42.78** +4.77%	**77.90** +4.07%	63.16 +1.17%	**70.53** +2.75%
Qwen3-4B	74.44	65.28	42.22	60.65	86.74	77.31	82.03
w/ GRPO	73.89	67.78	42.22	61.30	86.28	77.78	82.03
w/ OPSD	75.56	68.89	43.33	62.59	86.43	77.78	82.11
w/ OPSD+ReNIO	**75.83** +0.36%	**69.17** +0.41%	**46.67** +7.71%	**63.89** +2.08%	**87.04** +0.71%	**78.31** +0.68%	**82.68** +0.69%
Qwen3-8B	74.17	**71.67**	44.72	63.52	85.52	77.84	81.68
w/ GRPO	**78.33**	69.44	44.72	64.16	87.20	**78.51**	**82.86**
w/ OPSD	75.28	71.39	45.56	64.08	86.59	77.05	81.82
w/ OPSD+ReNIO	78.06 +3.69%	71.11 -0.39%	**47.50** +4.26%	**65.56** +2.31%	**88.41** +2.10%	77.31 +0.34%	**82.86** +1.27%
R1-Distill-Qwen-1.5B	27.78	22.78	14.44	21.67	49.09	41.07	45.08
w/ GRPO	**31.67**	24.17	13.89	**23.24**	49.24	41.47	45.36
w/ OPSD	26.66	23.89	13.06	21.20	**50.46**	40.87	45.67
w/ OPSD+ReNIO	28.61 +7.31%	**24.44** +2.30%	**15.28** +17.00%	22.78 +7.45%	50.30 -0.32%	**42.13** +3.08%	**46.22** +1.20%
R1-Distill-Qwen-7B	**56.11**	40.83	25.56	40.83	73.17	58.73	65.95
w/ GRPO	54.17	40.83	24.17	39.72	74.70	58.27	66.49
w/ OPSD	55.56	38.89	24.61	39.69	73.93	59.26	66.60
w/ OPSD+ReNIO	55.83 +0.49%	**42.78** +10.00%	**26.11** +6.09%	**41.57** +4.74%	**74.85** +1.24%	**59.39** +0.22%	**67.12** +0.78%

Table 2: Results of methods that do not require an additional teacher model. **Math Tasks**: AIME24, AIME25, and HMMT25 (Avg@12). **Coding Tasks**: HumanEval+ and MBPP+ (Avg@4). **Bold** denotes the best result per column within each model block. Subscripts on ReNIO rows report relative changes over the corresponding non-ReNIO row.



(a) Training time. (b) Trajectory weights. (c) Teacher entropy by weight.

Figure 4: (a) Training-time comparison between GRPO and prefix-based distillation. (b) Mean normalized weights of incorrect and correct trajectories during training. (c) Mean teacher entropy by trajectory-weight decile.

lation can apply dense teacher supervision before the final answer appears. Figure 4(a) shows the resulting cost difference: OPD and OPSD with 1024-token prefixes are substantially cheaper than GRPO, while extending the prefix length from 1024 to 4096, which usually covers the final answer in this setting, nearly triples the training time.We measure training time as the average wall-clock time per global step. All methods use the same effec- tive batch size, so the comparison remains fair even though GRPO uses more rollouts per prompt than OPD and OPSD.

Table 4 shows that this efficiency does not require a large loss in effectiveness, ReNIO further improves average performance under both prefix lengths. These results therefore support two conclusions: **short prefixes already provide effective OPD supervision, and ReNIO preserves this low-cost regime while improving it.**

Model	Math Tasks				Coding Tasks		
	AIME24	AIME25	HMMT25	Avg	HumanEval+	MBPP+	Avg
Teacher: Qwen3-8B							
Qwen3-1.7B	51.33	36.00	23.33	36.89	73.63	57.21	65.42
w/ OPD	54.44	39.16	27.50	40.37	74.40	62.43	68.42
w/ OPD+ReNIO	**54.72** +0.51%	**43.06** +9.96%	**28.33** +3.02%	**42.04** +4.14%	**75.00** +0.81%	**63.56** +1.81%	**69.28** +1.26%
Teacher: DS-R1-Qwen-7B							
DS-R1-Qwen-1.5B	27.78	22.78	14.44	21.67	49.09	41.07	45.08
w/ OPD	28.89	23.06	14.44	22.13	50.15	41.14	45.65
w/ OPD+ReNIO	**29.72** +2.87%	**23.89** +3.60%	**16.67** +15.44%	**23.43** +5.87%	**50.91** +1.52%	**41.20** +0.15%	**46.06** +0.90%

Table 3: Results of methods that require an additional teacher model.

Method	AIME24	AIME25	HMMT25	Avg
OPSD(1024)	53.06	41.94	27.50	40.83
/w ReNIO	**57.78**	42.78	**27.78**	**42.78**
OPSD(4096)	53.33	37.22	25.56	38.70
/w ReNIO	53.33	**43.06**	25.28	40.56
OPD(1024)	54.44	39.16	27.50	40.37
/w ReNIO	54.72	**43.06**	28.33	42.04
OPD(4096)	56.11	42.20	25.83	41.38
/w ReNIO	**56.39**	41.94	**28.61**	**42.31**

Table 4: Effect of SGO prefix length on Qwen3-1.7B OPSD and OPD mathematical reasoning performance. 1024 and 4096 denote the maximum training-prefix length; 4096 can almost cover the final answer of an SGO in our setting.

4.4 Analysis About Trajectory Weights During Training

ReNIO is designed to emphasize likely negative trajectories without using correctness labels. To verify whether the resulting trajectory weights exhibit this intended behavior, we partition the training SGOs according to final-answer correctness and track the mean normalized weight assigned to each group throughout training.

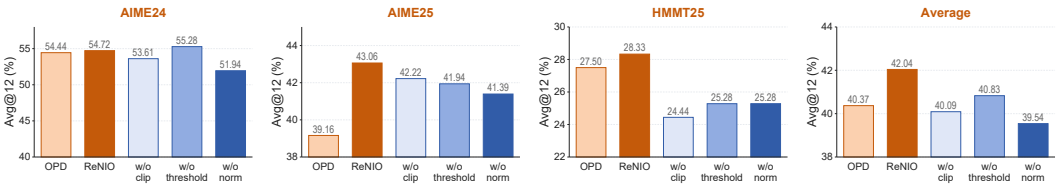
Figure 4(b) shows that ReNIO consistently redistributes training emphasis from correct SGOs toward incorrect SGOs under both distillation settings. This behavior supports the use of the prefix-computable student–teacher log ratio as a surrogate signal for negative-trajectory importance.

Another concern for ReNIO is that it may overweight low-quality SGOs produced by train–inference mismatch. If such trajectories fall outside the teacher's reliable region, their supervision could be weak or misleading. We examine this issue on Qwen3-1.7B using teacher entropy as a proxy for confidence: lower entropy indicates sharper and more reliable token-level guidance.

Figure 4(c) shows that higher ReNIO weights correlate with lower teacher entropy in both OPD and OPSD. This inverse relationship indicates that ReNIO does not primarily amplify trajectories for which the teacher distribution becomes uncertain. Instead, ReNIO assigns larger weights to SGOs where the student departs from the teacher but the teacher can still provide a sharp correction signal. We apply further analysis in Appendix D.1.

4.5 Ablation Studies

We apply ablation study on Qwen3-1.7B under OPD, using Qwen3-8B as the teacher. We study the three components used to construct the final ReNIO weight: key-token selection, log-ratio clipping, and batch-wise weight normalization. Detailed hyperparameter ablation studies are provided in Appendix D.4.The results are in Figure 5, and we can draw the following conclusions.



(a) AIME24. (b) AIME25. (c) HMMT25. (d) Average.

Figure 5: Component ablation of ReNIO under OPD on Qwen3-1.7B. Each panel compares OPD, full ReNIO, and ReNIO without log-ratio clipping (w/o clip), key-token selection (w/o threshold), or batch-wise normalization (w/o norm).

(1) **Key-token selection.** Removing threshold-based selection reduces the average score from 42.04 to 40.83, leaving a smaller improvement over OPD. This result supports selectively aggregating high-disagreement tokens when estimating trajectory importance. It is consistent with the log-ratio distribution in Figure 3: many tokens have log ratios near zero, so including them in the aggregation can dilute the contribution of the relatively few high-ratio tokens.

(2) **Log-ratio clipping.** Removing clipping lowers the average score to 40.09, a decrease of 1.95 percentage points relative to full ReNIO. This supports limiting the influence of extreme log ratios when constructing trajectory weights. Since the aggregated log ratio is exponentiated, unusually large values can disproportionately increase a trajectory's weight.

(3) **Batch-wise weight normalization.** Removing normalization produces the largest average decrease among the tested ablations, from 42.04 to 39.54. Normalization keeps the mean trajectory weight within each batch at one, controlling the overall weight scale while retaining relative differences between trajectories. The observed decrease supports this scale control under the evaluated training configuration.

5 Related Work

Classical knowledge distillation transfers teacher behavior to a student on fixed data or teacher-generated targets Hinton et al. (2015), but this offline setting can suffer from exposure bias in LLM post-training. On-policy distillation reduces this mismatch by sampling trajectories from the student policy and applying teacher supervision on the visited prefixes Agarwal et al. (2024; 2023); Lu & Lab (2025); Gu et al. (2023). Recent OPD studies cover self-distillation Zhao et al. (2026); Shenfeld et al. (2026); Hubotter et al. (2026); Kujanpää et al. (2024); Ye et al. (2026); Stein et al. (2026), mechanism analysis Kim et al. (2026); Zhang et al. (2026); Li et al. (2026); Yang et al. (2026), training stabilization Ko et al. (2026), and better use of student-generated outputs Xu et al. (2024); Ko et al. (2024; 2025); Lyu et al. (2025); Peng et al. (2025). Closest to our work, some methods reweight tokens or trajectories in OPD Wen et al. (2025); Zheng et al. (2026); Hou et al. (2026), but their weighting or calibration signals require knowing whether an SGO reaches the correct answer. ReNIO instead computes trajectory weights from prefix-conditioned student and teacher probabilities, preserving OPD's short-prefix advantage while emphasizing likely negative trajectories.

6 Conclusion

This paper shows that incorrect SGOs can provide especially useful supervision for on-policy distillation, motivating a training strategy that emphasizes negative trajectories without observing final-answer correctness. We propose ReNIO, a prefix-computable reweighting method that identifies pivotal tokens through student-to-teacher log ratios and aggregates them into normalized trajectory weights. Because ReNIO does not require the answer of an SGO for weighting, it preserves OPD's advantage over RL by allowing training on SGO prefixes. Across mathematical reasoning and code generation tasks, ReNIO improves both OPD and OPSD while preserving short-prefix training.

LIMITATIONS

We evaluate ReNIO across multiple model families and task domains, showing consistent gains on mathematical reasoning and code generation. However, due to hardware constraints, we have not verified its effectiveness on larger-scale models.

ETHICS STATEMENT

Our experiments use publicly available models, public benchmark datasets, and model-generated on-policy trajectories. We do not collect private user data, conduct human-subject studies, or use personally identifiable information. To the best of our knowledge, our use of these resources follows their intended research purposes and does not introduce known intellectual-property, ethical, or legal disputes.

USE OF AI ASSISTANTS

We use AI assistants to help enrich our writing and finish experiments.

REPRODUCIBILITY STATMENTS

To support reproducibility, we provide code and detailed guidelines for running the experiments. We will include the code, and relevant experimental resources in a supplementary package submitted through OpenReview. Upon acceptance, we will publicly release all relevant materials on GitHub.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stańczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, 2023. URL https://api.semanticscholar.org/CorpusID:263610088.

Rishabh Agarwal, Nino Vieillard, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *Proceedings of the 12th International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Kimi Team Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Haochen Ding, Meng xiao Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, Yichen Feng, Kelin Fu, Bofei Gao, Hongcheng Gao, Peizhong Gao, Tong Gao, Xinran Gu, Longyu Guan, Haiqing Guo, Jianhang Guo, Hao-Xing Hu, Xiaoru Hao, Tian He, Weiran He, Wen He, Chao Hong, Yan-Ni Hu, Zhenxing Hu, Weixiao Huang, Zhiqi Huang, Zihao Huang, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yongsheng Kang, Guokun Lai, Cheng Li, Fang Li, Haoyang Li, Ming Li, Wentao Li, Yanhao Li, Yiwei Li, Zhaowei Li, Zheming Li, Hong-Li Lin, Xiaohan Lin, Zongyu Lin, Chengyi Liu, Chenyu Liu, Hongzhang Liu, Jingyuan Liu, Junqi Liu, Liang Liu, Shaowei Liu, T. Y. Liu, Tian-Bo Liu, Weizhou Liu, Yangyang Liu, Yibo Liu, Yiping Liu, Yue Liu, Zhengying Liu, Enzhe Lu, Li Lu, Shen Ma, Xinyu Ma, Yi-Xuan Ma, Shaoguang Mao, Jie Mei, Xin Men, Yibo Miao, Siyuan Pan, Yebo Peng, Ruoyu Qin, Bowen Qu, Zeyu Shang, Li-Na Shi, Sheng-Rong Shi, Feifan Song, Jian-Fei Su, Zhen-Xin Su, Xinjie Sun, Flood Sung, Heyi Tang, Ji-Hua Tao, Qi Teng, Chensi Wang, Dinglu Wang, Feng Wang, Haiming Wang, Jianzhou Wang, Jiaxing Wang, Jinhong Wang, Shengjie Wang, Shuyi Wang, Yao Wang, Yejie Wang, Yiqin Wang, Yuxin Wang, Yuzhi Wang, Zhaoji Wang, Zhengtao Wang, Zhexu Wang, Chu Wei, Qi-Feng Wei, Wenhao Wu, Xingzhe Wu, Yuxin Wu, Chenjun Xiao, Xiao-Ming Xie, Weiming Xiong, Boyu Xu, Jing Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Zi-Yang Xu, Junjie Yan, Yuzi Yan, Xiaofei Yang, Ying Yang, Zhengqi Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Xingcheng Yao, Wen guang Ye, Zhuorui Ye, Bohong Yin, Long Yu, Enming Yuan, Hongbang Yuan, Mengjie Yuan, Haobing Zhan, Dehao Zhang, Hao Zhang, Wanlu Zhang, Xiaobin Zhang, Yangkun Zhang,

Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Hao-Dong Zhao, Yikai Zhao, Huabin Zheng, Shao Jian Zheng, Jianren Zhou, Xinyu Zhou, Zaida Zhou, Zhengxin Zhu, Weiyu Zhuang, and Xinxing Zu. Kimi k2: Open agentic intelligence. 2025. URL https://api.semanticscholar.org/CorpusID:280323540.

Kedi Chen, Dezhao Ruan, Yuhao Dan, Yaoting Wang, Siyu Yan, Xuecheng Wu, Yinqi Zhang, Qin Chen, Jie Zhou, Liang He, Biqing Qi, Linyang Li, Qipeng Guo, Xiaoming Shi, and Wei Zhang. A survey of inductive reasoning for large language models, 2026. URL https://arxiv.org/abs/2510.10182.

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

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Jun-Mei Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiaoling Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bing-Li Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dong-Li Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, JingChang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, M. Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, Rui-Qi Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shao-Kang Wu, Tao Yun, Tian Pei, T. Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wen-Xia Yu, Wentao Zhang, Wangding Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyu Jin, Xi-Cheng Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yi Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yu-Jing Zou, Yujia He, Yunfan Xiong, Yu-Wei Luo, Yu mei You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yao Li, Yi Zheng, Yuchen Zhu, Yunxiang Ma, Ying Tang, Yukun Zha, Yuting Yan, Zehui Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhen guo Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zi-Long Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633 – 638, 2025. URL https://api.semanticscholar.org/CorpusID:275789950.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. 2026. URL https://arxiv.org/abs/2605.00674.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: On-policy distillation of large language models. 2023. URL https://api.semanticscholar.org/CorpusID:259164722.

Etash Kumar Guha, Ryan Marten, Sedrick Scott Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean-Pierre Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Ben Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal

Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models. *ArXiv*, abs/2506.04178, 2025. URL https://api.semanticscholar.org/CorpusID:279154475.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. URL https://arxiv.org/abs/1503.02531.

Wenjin Hou, Shangpin Peng, Weinong Wang, Zheng Ruan, Yue Zhang, Zhenglin Zhou, Mingqi Gao, Yifei Chen, Kaiqi Wang, Hongming Yang, Chengquan Zhang, Zhuotao Tian, Han Hu, Yi Yang, Fei Wu, and Hehe Fan. Uni-opd: Unifying on-policy distillation with a dual-perspective recipe, 2026. URL https://arxiv.org/abs/2605.03677.

Jonas Hubotter, Frederike Lubeck, Lejs Deen Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. *ArXiv*, abs/2601.20802, 2026. URL https://api.semanticscholar.org/CorpusID:285102353.

Zixuan Ke, Fangkai Jiao, Yifei Ming, Xuan-Phi Nguyen, Austin Xu, Do Xuan Long, Minzhi Li, Chengwei Qin, Peifeng Wang, Silvio Savarese, Caiming Xiong, and Shafiq Joty. A survey of frontiers in llm reasoning: Inference scaling, learning to reason, and agentic systems, 2026. URL https://arxiv.org/abs/2504.09037.

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of llms? 2026. URL https://api.semanticscholar.org/CorpusID:286776340.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models. In *Forty-first International Conference on Machine Learning*, 2024.

Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. Distillm-2: A contrastive approach boosts the distillation of llms. *arXiv preprint arXiv:2503.07067*, 2025.

Jongwoo Ko, Sara Abdali, Young Jin Kim, Tianyi Chen, and Pashmina Cameron. Scaling reasoning efficiently via relaxed on-policy distillation. 2026. URL https://api.semanticscholar.org/CorpusID:286489350.

Kalle Kujanpää, Pekka Marttinen, Harri Valpola, and Alexander Ilin. Efficient knowledge injection in llms via self-distillation. *Trans. Mach. Learn. Res.*, 2025, 2024. URL https://api.semanticscholar.org/CorpusID:274859956.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026. URL https://arxiv.org/abs/2604.13016.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=1qvx610Cu7.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Yuanjie Lyu, Chengyu Wang, Jun Huang, and Tong Xu. From correction to mastery: Reinforced distillation of large language model agents. *ArXiv*, abs/2509.14257, 2025. URL https://api.semanticscholar.org/CorpusID:281393943.

Jingyu Peng, Maolin Wang, Hengyi Cai, Yuchen Li, Kai Zhang, Shuaiqiang Wang, Dawei Yin, and Xiangyu Zhao. Adaswitch: Balancing exploration and guidance in knowledge distillation via adaptive switching. 2025. URL https://api.semanticscholar.org/CorpusID:286579337.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *ArXiv*, abs/2601.19897, 2026. URL https://api.semanticscholar.org/CorpusID:285071839.

Alex Stein, Furong Huang, and Tom Goldstein. Gates: Self-distillation under privileged context with consensus gating. *ArXiv*, abs/2602.20574, 2026. URL https://api.semanticscholar.org/CorpusID:286001285.

Xumeng Wen, Zihan Liu, Shun Zheng, Zhijian Xu, Shengyu Ye, Zhirong Wu, Xiao Liang, Yang Wang, Junjie Li, Ziming Miao, Jiang Bian, and Mao Yang. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *ArXiv*, abs/2506.14245, 2025. URL https://api.semanticscholar.org/CorpusID:279410727.

Wenda Xu, Rujun Han, Zifeng Wang, Long T. Le, Dhruv Madeka, Lei Li, William Yang Wang, Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. *ArXiv*, abs/2410.11325, 2024. URL https://api.semanticscholar.org/CorpusID:273350649.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Jingren Zhou, Junyan Lin, Kai Dang, Keqin Bao, Ke-Pei Yang, Le Yu, Li-Chun Deng, Mei Li, Min Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shi-Qiang Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. 2025. URL https://api.semanticscholar.org/CorpusID:278602855.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *ArXiv*, abs/2602.12125, 2026. URL https://api.semanticscholar.org/CorpusID:285540530.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *ArXiv*, abs/2602.12275, 2026. URL https://api.semanticscholar.org/CorpusID:285540636.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Honglin Yu, Weinan Dai, Yuxuan Song, Xiang Wei, Haodong Zhou, Jingjing Liu, Wei Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yong-Xu Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale. *ArXiv*, abs/2503.14476, 2025. URL https://api.semanticscholar.org/CorpusID:277104124.

Dan Zhang, Zhuohan Yang, Sepehr Janghorbani, Jiaqi Han, Andrew Ressler, Qi Qian, Gabriel D. Lyng, Sarthak S. Batra, and Richard E. Tillman. Fast and effective on-policy distillation from reasoning prefixes. *Manuscript*, 2026.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2024, 2024.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *ArXiv*, abs/2601.18734, 2026. URL https://api.semanticscholar.org/CorpusID:285050713.

Binbin Zheng, Xing Ma, Yiheng Liang, Jingqing Ruan, Xiaoliang Fu, Kepeng Lin, Benchang Zhu, Ke Zeng, and Xunliang Cai. Scope: Signal-calibrated on-policy distillation enhancement with dual-path adaptive weighting, 2026. URL https://arxiv.org/abs/2604.10688.

Qihao Zhu Runxin Xu Junxiao Song Mingchuan Zhang Y.K. Li Y. Wu Daya Guo Zhihong Shao, Peiyi Wang. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *ArXiv*, abs/2506.01347, 2025. URL https://api.semanticscholar.org/CorpusID:279075301.

A Reverse-KL Gradient Interpretation of the Student–Teacher Ratio

This appendix derives why the student-to-teacher log ratio used by ReNIO can be interpreted as a token-level gradient weight under reverse-KL distillation. Consider one fixed prefix $(x, y_{<t})$ and write the student and teacher next-token distributions as $p_S(v)$ and $p_T(v)$. The reverse-KL objective at this prefix is

$$\begin{aligned}\mathcal{L}_{\mathrm{RKL}} &= \mathrm{KL}(p_S \| p_T) \\ &= \sum_v p_S(v)\big(\log p_S(v) - \log p_T(v)\big).\end{aligned} \tag{9}$$

Let z_u be the student logit for token u, so that $p_S(v) = \mathrm{softmax}(z)_v$. Differentiating Eq. 9 with respect to z_u gives

$$\begin{aligned}\frac{\partial \mathcal{L}_{\mathrm{RKL}}}{\partial z_u} = \sum_v \frac{\partial p_S(v)}{\partial z_u} \\ \big(\log p_S(v) - \log p_T(v) + 1\big).\end{aligned} \tag{10}$$

Using the softmax derivative $\partial p_S(v)/\partial z_u = p_S(v)(\mathbb{I}[v = u] - p_S(u))$, we obtain

$$\begin{aligned}\frac{\partial \mathcal{L}_{\mathrm{RKL}}}{\partial z_u} &= p_S(u)\big(\log p_S(u) - \log p_T(u) + 1\big) \\ &\quad - p_S(u) \sum_v p_S(v)\big(\log p_S(v) - \log p_T(v) + 1\big).\end{aligned} \tag{11}$$

The second term is shared across all vocabulary tokens at the same prefix and acts as a distribution-level baseline. The token-specific part is therefore controlled by $\log p_S(u) - \log p_T(u) + 1 = 1 + \log \frac{p_S(u)}{p_T(u)}$. Equivalently, the same result can be written in score-function form:

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\mathrm{RKL}} = \mathbb{E}_{v \sim p_S}\big[&(\log p_S(v) - \log p_T(v) + 1) \\ &\cdot \nabla_\theta \log p_S(v)\big].\end{aligned} \tag{12}$$

Because $\mathbb{E}_{v \sim p_S}[\nabla_\theta \log p_S(v)] = \nabla_\theta \sum_v p_S(v) = 0$, the constant $+1$ is a removable baseline. Therefore,

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\mathrm{RKL}} = \mathbb{E}_{v \sim p_S}\big[&(\log p_S(v) - \log p_T(v)) \\ &\cdot \nabla_\theta \log p_S(v)\big],\end{aligned} \tag{13}$$

so the effective token-level weight is exactly the student-to-teacher log ratio.

For an on-policy SGO, the observed token y_t is sampled from the student distribution. Substituting $v = y_t$ into Eq. 13 shows that the sampled token's effective reverse-KL gradient contribution is weighted by

$$\ell_t = \log \frac{p_S(y_t \mid x, y_{<t})}{p_T(y_t \mid x, y_{<t})}. \tag{14}$$

Thus, tokens that the student assigns much higher probability than the teacher receive larger reverse-KL corrective emphasis. ReNIO uses this same log-ratio term as token-level evidence for identifying pivotal decisions and then aggregates the selected evidence into a normalized trajectory-level weight.

B Detailed Experimental Setup

Here we list the detailed hyperparameter values for each setting in Section 4. We provide the training and evaluation configurations for GRPO, OPD, OPSD experiments in Tables 5, 6, 7. Following the settings in OPSD Zhao et al. (2026), for Qwen3 series models' OPD and OPSD training, we apply Thinking-Mode-off student/Thinking-Mode-on teacher, and for their evaluation, we apply Thinking-Mode on.

All experiments are applied on 4 H200 GPUs with gradient checkpointing and Flash Attention 2. We utilize the AdamW optimizerLoshchilov & Hutter (2019) and bfloat16 precision for all training runs.And we apply full-vocabulary logit distillation for all OPD and OPSD experiments.

Parameter	GRPO	OPD	OPSD
Learning rate	5e-6	5e-6	5e-6
Effective batch size	32	32	32
LoRA rank (r)	64	64	64
LoRA alpha (α)	128	128	128
LoRA target modules	q proj, k proj, v proj, o proj, gate proj, up proj, down proj		
Max completion length	16000	1024	1024
Number of generations per prompt	8	1	1
Sampling temperature	1.2	1.1	1.1
Training steps	300	100	100

Table 5: Training configuration for Qwen3 series models.

Parameter	GRPO	OPD	OPSD
Learning rate	1e-6	1e-6	1e-6
Effective batch size	32	32	32
LoRA rank (r)	64	64	64
LoRA alpha (α)	128	128	128
LoRA target modules	q proj, k proj, v proj, o proj, gate proj, up proj, down proj		
Max completion length	16000	2048	2048
Number of generations per prompt	8	1	1
Sampling temperature	1.2	1.1	1.1
Training steps	300	100	100

Table 6: Training configuration for DS-Distill-Qwen series models.

Parameter	Math	Coding
Max New Tokens	38912	38912
Thinking Mode	Enabled	Enabled
Temperature	1.0	1.0
Top-p	0.95	0.95
Top-k	-1	-1
Min-p	0.0	0.0
Presence Penalty	0.0	0.0
Samples per Prompt	12	4

Table 7: Evaluation configuration

Additionally, for experiments conducted in Section 2, At each training step, we generate SGOs on policy, label them by final-answer correctness, and retain only one correctness group for the on-policy distillation update. We train separate models under the same OPD or pipeline. All other training data and optimization settings are kept unchanged, so the comparison isolates the effect of using correct versus incorrect on-policy trajectories. We evaluate each trained model with Avg@12 on mathematical reasoning benchmarks.

We keep most training and evaluation settings the same as those in Tables 5 and 7, except for the maximum completion length. Since we need the final output to identify whether the SGO is correct, we must roll out the whole trajectory. For Qwen3-1.7B with Thinking Mode off, a generation length of 4096 is enough to generate the final answer; hence, we set the maximum generation length to 4096 for experiments in Section 2.

C ReNIO Algorithm

Algorithm 1: **ReNIO** Algorithm for On-Policy Distillation

Require: Student S, teacher T, on-policy batch $\mathcal{B} = \{(x_i, y_i)\}_{i=1}^{B}$, threshold τ, clip bounds $\epsilon_{\min}, \epsilon_{\max}$
Ensure: Normalized trajectory weights $\{\hat{w}_i\}_{i=1}^{B}$

for each trajectory $(x_i, y_i) \in \mathcal{B}$ **do**
  Phase I: Student-Teacher Log Ratio
  for each token position $t = 1, \ldots, |y_i|$ **do**
    $\ell_t \leftarrow \log p_S(y_t \mid x_i, y_{<t}) - \log p_T(y_t \mid x_i, y_{<t})$
  end for
  Phase II: Key Token Information Selection
  $\mathcal{K}_i \leftarrow \emptyset$
  for each token position $t = 1, \ldots, |y_i|$ **do**
    if $\ell_t > \tau$ **then**
      $\mathcal{K}_i \leftarrow \mathcal{K}_i \cup \{t\}$
    end if
  end for
  Phase III: Token Information Aggregation
  if $\mathcal{K}_i \neq \emptyset$ **then**
    $\bar{\ell}_i \leftarrow \frac{1}{|\mathcal{K}_i|} \sum_{t \in \mathcal{K}_i} \mathrm{clip}(\ell_t,\ \epsilon_{\max})$
    $w_i \leftarrow \exp(\bar{\ell}_i)$
  else
    $w_i \leftarrow 1$ ▷ no pivotal token found
  end if
end for
$\bar{w}_B \leftarrow \frac{1}{B} \sum_{i=1}^{B} w_i$
for $i = 1, \ldots, B$ **do**
  $\hat{w}_i \leftarrow w_i \,/\, \bar{w}_B$
end for
return $\{\hat{w}_i\}_{i=1}^{B}$

D Additional Analyses and Discussions

D.1 Supplementary Analysis On Entropy-Weight

The pattern shown in 4c is plausible because current student models are already capable enough that an incorrect trajectory does not necessarily collapse into incoherent reasoning. As illustrated in Figure 2b, the student chooses the wrong pivotal token "13", yet the subsequent reasoning still follows a structured path conditioned on that mistaken branch. Such trajectories remain within a region where the teacher can recognize the local error with high confidence: the student assigns high probability to the sampled token, while the teacher assigns it low probability, producing a large student-to-teacher ratio and therefore a large ReNIO weight.

The opposite case explains the low-weight, high-entropy region in Figure 4(c). If a student trajectory is far outside the teacher's reliable reasoning distribution, the teacher distribution becomes flatter and less decisive. In that case, the denominator in the student-to-teacher ratio is no longer

sharply suppressed for a particular wrong token, so the ratio may become smaller rather than larger. Extremely erroneous trajectories therefore receive small ReNIO weights and are not strongly amplified. Overall, the negative correlation between ReNIO weight and teacher entropy supports the intended behavior of the method: **high weights correspond to student errors on which the teacher remains confident enough to give clear supervision. This further validates the Rationality of ReNIO.**

D.2 SUPPLEMENTARY ANALYSIS ON WEIGHTING ALGORITHM

To further examine the design of ReNIO, we compare it with two alternative weighting strategies under Qwen3-1.7B OPSD. The first variant uses the same student-to-teacher ratio signal as ReNIO but applies the resulting weights directly to token losses instead of aggregating them into a trajectory-level weight. The second variant keeps ReNIO's trajectory-level weighting form but reverses the information signal, using the teacher-to-student ratio instead of the student-to-teacher ratio.

Method	AIME24	AIME25	HMMT25	Avg
Base	51.33	36.00	23.33	36.89
OPSD	53.06	41.94	27.50	40.83
/w ReNIO	**57.78**	**42.78**	27.78	**42.78**
/w T/S trajectory weighting	55.28	40.00	**29.44**	41.57
/w S/T token weighting	54.72	40.28	25.83	40.28

Table 8: Comparison of alternative weighting strategies under Qwen3-1.7B OPSD. Scores are Avg@12 on AIME24, AIME25, and HMMT25. S/T token weighting applies student-to-teacher ratio weights at the token level; T/S trajectory weighting keeps trajectory-level aggregation but uses the reverse teacher-to-student ratio.

Table 8 shows that directly applying the S/T signal at the token level is ineffective: its average score is even lower than the OPSD baseline. This indicates that the S/T ratio should not be used to rescale every token loss independently; instead, it is more useful after being aggregated into a trajectory-level signal, as in ReNIO. The T/S trajectory-weighting variant improves over OPSD, but still underperforms ReNIO by 1.21 points. A likely explanation is that high T/S ratios emphasize tokens that the teacher assigns high probability but the student assigns low probability; such tokens may occur more often in correct or teacher-aligned SGOs, giving larger weights to trajectories that are not the main source of the negative-trajectory signal identified in Section 2. This supports ReNIO's design choice of using S/T ratios to emphasize likely negative, high-information SGOs.

D.3 COMPARISON WITH CORRECTNESS-AWARE WEIGHTING METHOD

We compare ReNIO with SCOPE, which is also a trajectory reweighting method, under an identical maximum generation length of 4096, which is sufficient to produce answer-containing trajectories in our setting. SCOPE explicitly uses final-answer correctness to route correct and incorrect trajectories to student-perplexity-weighted maximum-likelihood estimation and teacher-perplexity-weighted KL distillation, respectively. ReNIO receives the same rollout budget but computes its weights from student–teacher probabilities without using correctness labels.

Table 9 shows that ReNIO attains a higher three-benchmark average in all four configurations. Because both methods use 4096 tokens here, this table does not measure rollout savings. However, SCOPE must wait for final-answer correctness, whereas ReNIO's prefix-computable weights can be applied before trajectory completion. ReNIO therefore remains compatible with the lower-cost truncated-prefix regime evaluated in the main paper.

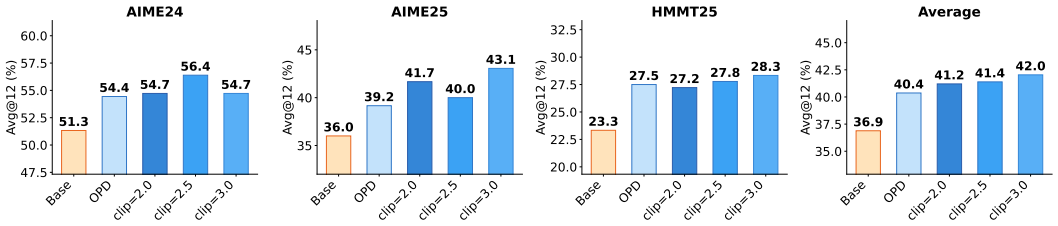
D.4 SUPPLEMENTAR ABLATION STUDIES

We vary the two main hyperparameters: the clipping bound and the key-token threshold. Following Section 4.5, we conduct this experiment with OPD on Qwen3-1.7B. In each sweep, the other hyperparameter is fixed to its default value. Figure 6 summarizes the per-benchmark results across the three mathematical reasoning benchmarks.

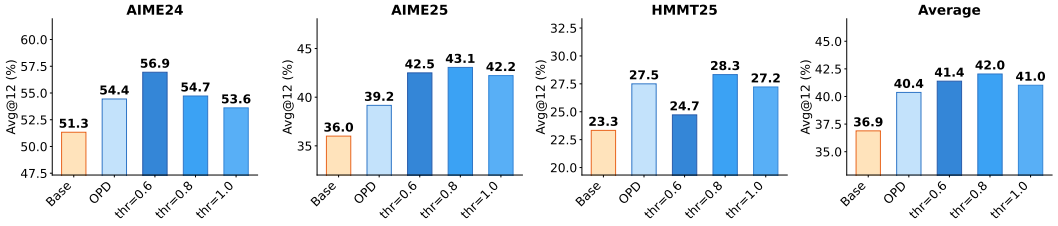
Method	R1-Distill-Qwen-1.5B				Qwen3-1.7B			
	AIME24	AIME25	HMMT25	Avg	AIME24	AIME25	HMMT25	Avg
OPSD+SCOPE	**30.28**	23.33	14.17	22.59	48.61	38.61	25.00	37.41
OPSD+ReNIO	30.00	**24.17**	**15.00**	**23.06**	**53.33**	**43.06**	**25.28**	**40.56**
OPD+SCOPE	28.61	24.72	13.33	22.22	50.00	36.11	23.61	36.57
OPD+ReNIO	**31.39**	**25.28**	**13.89**	**23.52**	**56.39**	**41.94**	**28.61**	**42.31**

Table 9: Comparison of ReNIO and SCOPE under a matched maximum generation length of 4096. The Qwen3-1.7B student is evaluated with two teachers. Scores are Avg@12 on AIME24, AIME25, and HMMT25; Avg is their arithmetic mean. T denotes the teacher model. **Bold** denotes the better result within each OPD or OPSD pair.

The hyperparameter ablations show the same stability–sparsity pattern. A small clipping bound suppresses useful disagreement, while a loose bound gives too much influence to extreme ratios; the best average is obtained at a clipping bound of 3.0. For key-token selection, a low threshold admits routine tokens and dilutes the signal, whereas a high threshold discards informative disagreements; the best average is obtained at a threshold of 0.8. Together, these results support the design of ReNIO: it emphasizes salient student–teacher disagreements while keeping trajectory weights numerically stable.



(a) Effect of the clipping bound on OPD. The threshold is fixed to 0.8.



(b) Effect of the key-token threshold on OPD. The clipping bound is fixed to 3.0.

Figure 6: Per-benchmark hyperparameter ablation results for ReNIO under OPD on Qwen3-1.7B mathematical reasoning.

E Additional Experimental Results

E.1 Supplementar Motivation Experiment Results

Here we provide the full experimental results for Section 2, as shown in Figures 7.

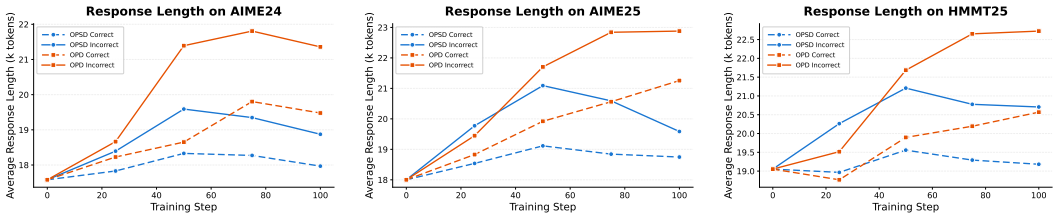


Figure 7: Average response length of correct versus incorrect on-policy trajectories during training on AIME24 (left), AIME25 (middle), and HMMT25 (right). Incorrect trajectories are consistently longer, and the gap widens over training steps, indicating that incorrect trajectories accumulate more deviation tokens.

F Qualitative Comparison

Question: Find the number of triples of nonnegative integers (a, b, c) satisfying $a + b + c = 300$ and

$$a^2b + a^2c + b^2a + b^2c + c^2a + c^2b = 6{,}000{,}000.$$

Ground-truth answer: $\boxed{601}$

[GRPO]

<think> Okay, so I need to find the number of triples of nonnegative integers (a, b, c) satisfy $a^2b + a^2c + b^2a + b^2c + c^2a + c^2b = 6{,}000{,}000$
...So that simplifies the second equation to: $a^2(300 - a) + b^2(300 - b) + c^2(300 - c) = 6{,}000{,}000$
this is satisfied for any $(100, x, 200 - x)$, which is 201 solutions. Therefore, the answer is 201.
</think>
Wrong answer: $\boxed{201}$
Error. This response only counts the family with $a = 100$. It misses the symmetric families where $b = 100$ or $c = 100$, so it undercounts the ordered triples.

[OPD]

<think> Okay, so I need to find the number of triples of nonnegative integers (a, b, c) such that a + b + c = 300...
the actual **number of ordered triples** where **exactly one variable is 100** and the other two sum to 200 is: 201 total unordered (b, c) pairs that result in $b + c = 200$...
...If we consider all ordered triples where one variable is 100 and the other two are nonnegative integers summing to 200, then the number of such ordered triples is 3 * 201. Because for each of the three positions where the 100 can be placed, and for each such position, there are 201 ordered pairs (b, c).
Hence, there are $3 * 201 = 603$ **ordered triples**.
</think>
Wrong answer: $\boxed{603}$
Error. This response counts all three symmetric families but does not remove their overlap.

[OPD+ReNIO]

<think> the number of solutions is the number of nonnegative integer solutions to $b + c = 200$, which is 201 (from $b = 0$ to $b = 200$).
...for each of the three positions where 100 can be, and for each of the 201 possible pairs (b, c), there are 3 positions for 100.
But wait, if we fix $a = 100$, then $b + c = 200$, which gives 201 solutions. Similarly, if we fix $b = 100$, then $a + c = 200$, giving another 201 solutions, and similarly for $c = 100$, giving another 201 solutions. This would lead to $3 * 201 = 603$ solutions.
...
Alternatively, since each of the three sets ($a = 100$, $b = 100$, $c = 100$) includes the triple $(100, 100, 100)$ once. Therefore, the total number of distinct ordered triples is $3 * 201 - 2 = 603 - 2 = 601$.Therefore, the number of triples is 601.
</think>
Correct answer: $\boxed{601}$
Correct. This response preserves both necessary steps.