

Joint Training of Multi-Token Prediction in Reinforcement Learning via Optimal Coefficient Calibration

Zili Wang^{1,2*}, Jiajun Chai³, Lin Chen^{1,2}, Xiaohan Wang³,
Shiming Xiang^{1,2}, Guojun Yin³

¹ School of Artificial Intelligence, University of Chinese Academy of Sciences, China

² MAIS, Institute of Automation, Chinese Academy of Sciences, China ³ Meituan

 Code: <https://github.com/MarkXCloud/RL-MTP>

Abstract

Reinforcement Learning from Verifiable Rewards (RLVR) has emerged as the standard paradigm for improving reasoning capability of large language models, while Multi-Token Prediction (MTP) has been a widely adopted module in pretraining. Combining them is a natural approach, yet current RL practices detach MTP gradients because joint training degrades the performance. We revisit this failure from an optimization perspective. We show that the per-step effect of MTP on the RL objective can be decomposed into two terms: a first-order correlation and a second-order perturbation penalty. This decomposition unifies three MTP training regimes: Detach, Cross-Entropy loss, and Policy loss, and explains why each succeeds or fails. Further analysis of policy loss reveals that, although it aligns with intuition, performance still degrades: the correlation term decays while the quadratic penalty persists. Guided by the analysis, we propose **Optimal Coefficient Calibration (OCC)**, an adaptive scheme that tracks the optimal coefficient online via a log-probability proxy at negligible cost. Across six competition-level mathematical reasoning benchmarks, OCC consistently matches or exceeds the detach baseline, delivering improved joint MTP-RL training performance.

1 Introduction

Reinforcement Learning from Verifiable Rewards (RLVR) has become the standard paradigm for enhancing the reasoning capability of Large Language Models (LLMs) (Schulman et al., 2017; Shao et al., 2024; Yu et al., 2025) during post-training. Concurrently, Multi-Token Prediction (MTP) (Gloeckle et al., 2024; Stern et al., 2018; Liu et al., 2024), which is trained to predict multiple future tokens after the main model, has demonstrated substantial benefits during pretraining, including

capturing multi-step representations, improving downstream accuracy and serving as a native draft model for speculative decoding (Leviathan et al., 2023; Chen et al., 2023). Given these strengths, an intuitive approach is to *jointly* optimize MTP and RL objectives during post-training, with the expectation that the model may benefit from multi-token supervision while acquiring reasoning abilities.

However, *previous joint MTP-RL training shows significant performance degradation on the main model*, as shown in Figure 1. Therefore, previous practices choose to detach MTP head as a compromise. Widely adopted RL frameworks (e.g., veRL (Sheng et al., 2025), slime (Zhu et al., 2025)) explicitly report the degradation and recommend gradient detachment (veRL Team, 2026; Zhao, 2024). This has pushed recent prominent models toward similar choices. GLM-5 (GLM-5-Team et al., 2026) and Composer-2 (Research et al., 2026) isolate MTP head during RL training. Nemotron-3 Super (NVIDIA, 2025) freezes the main model after RL convergence and fine-tunes MTP head separately. When MTP is detached, its parameters can only adapt to the main model through an independent post-hoc fine-tuning procedure, sacrificing the potential gains from multi-token supervision.

This leads to a natural question: *Why does joint MTP-RL training fail, and can we design a principled strategy to make it work?*

In this paper, we first analyze the joint MTP-RL training from an optimization perspective. We decompose the per-step effect of MTP into: (a) a *first-order correlation term*, determined by the directional agreement between RL and MTP gradients, and (b) a *second-order penalty term*, reflecting the perturbation introduced by MTP gradients. Performance improvement depends on whether the correlation term outweighs the penalty term.

This decomposition unifies three MTP training regimes: (1) **Detach** isolates the MTP gradient from the main model, same as training main model

*Work done as an intern at Meituan.

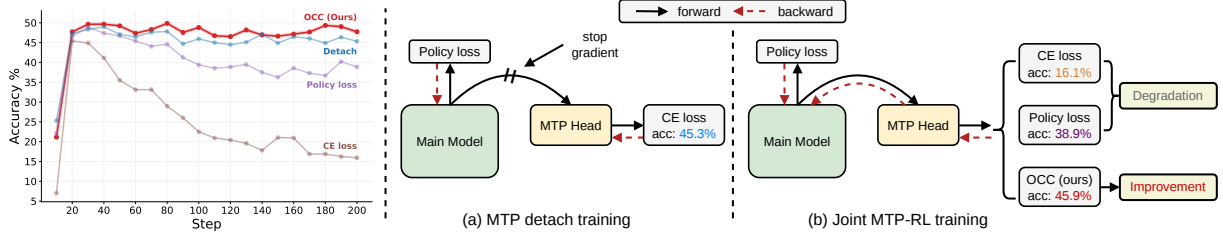


Figure 1: **Left:** Accuracy on AIME24. CE loss shows great degradation, while policy loss initially surpasses but degrades as training progresses. In contrast, OCC sustains gains throughout training and consistently outperforms. **Middle & Right:** Illustrations of the MTP training regimes. Detach training stops the gradient from MTP into the main model. Joint MTP-RL training allows this gradient to flow back into the main model, enabling joint updates.

alone. (2) **Cross-Entropy (CE) Loss** treats all samples equally with cross-entropy loss, whereas RL up-weights high-reward samples and suppresses low-reward ones; the unrelated objective yields a weak correlation term that cannot outweigh the perturbation. (3) **Policy Loss** uses the same RL objective as the main model, so gradients are initially well aligned and the correlation term dominates; however, as the gradient approaches a flat region (Li et al., 2018), this alignment decays while the perturbation persists, causing a rise-then-fall performance curve. A fixed MTP coefficient cannot track this phase transition, motivating an adaptive coefficient to calibrate the drift throughout training.

Based on this analysis, we propose **Optimal Coefficient Calibration (OCC)**, an adaptive procedure that calibrates the optimal MTP coefficient during training. OCC starts from the closed-form optimum implied by our analysis. To avoid computing full-model gradient in large-scale distributed training system, we use a log-probability proxy derived from the log probability change under a small-step approximation (Li et al., 2026; Ma et al., 2026). This proxy allows OCC to track the theoretically preferred coefficient online with negligible computational overhead.

We evaluate on multiple competition-level mathematical reasoning benchmarks with different models and algorithms compared with the three training regimes. Extensive results show that cross-entropy loss underperforms the Detach. Policy loss exhibits a rise-then-fall trend. In contrast, our OCC consistently matches or exceeds the Detach across all benchmarks, demonstrating stable improvements across tasks.

Our contributions are summarized as follows:

1. We provide a theoretical analysis of joint MTP-RL training. This analysis unifies three MTP training regimes and explains why each

succeeds or fails.

2. We propose **Optimal Coefficient Calibration (OCC)** that adaptively calibrates the MTP coefficient online using a log-probability proxy, requiring negligible computational overhead and no full-model gradient computation.
3. We conduct extensive experiments on multiple competition-level mathematical reasoning benchmarks across different models and algorithms. Results demonstrate that OCC consistently matches or exceeds the Detach, demonstrating stable improvements across tasks.

2 Related Work

Multi-Token Prediction. The concept of predicting multiple future tokens as an auxiliary objective was formalized by Gloeckle et al. (2024), who demonstrated that training LLMs to simultaneously predict k future tokens improves sample efficiency and downstream task performance. This idea has since been adopted at scale: DeepSeek-V3&V4 (Liu et al., 2024; DeepSeek-AI, 2026), LongCat-Flash (Team, 2025a) and StepFun-3.5-Flash (Huang et al., 2026) incorporate MTP with cross-entropy supervision. MiMo (Xiaomi, 2025) and Qwen3-Next (Team, 2025b) employ similar multi-token heads. On the efficiency side, FastMTP (Cai et al., 2025) aligns MTP training with its recursive inference pattern, achieving significant speedups through self-distillation and dynamic vocabulary compression. The MTP heads can serve as draft models for speculative decoding (Leviathan et al., 2023; Chen et al., 2023) at inference time. These stronger representations plus faster inference make MTP an increasingly standard component in modern LLM architectures.

Reinforcement Learning Post-Training. Reinforcement Learning from Verifiable Rewards

(RLVR) has become the standard paradigm for enhancing the reasoning capabilities of LLMs. Proximal Policy Optimization (PPO) (Schulman et al., 2017) established the foundation, using clipped surrogate objectives with a value function baseline and KL regularization against a reference policy. Group relative policy optimization (GRPO) (Shao et al., 2024) simplifies this by eliminating the critic network and computing advantages via group-level reward normalization. Decoupled clip and dynamic sampling policy optimization (DAPO) (Yu et al., 2025) further refines the paradigm with clip-higher, dynamic sampling, and overlong reward shaping to improve training stability on mathematical reasoning tasks. Group sequence policy optimization (GSPO) (Zheng et al., 2025) reduces variance during large-scale training by shifting the optimization objective from tokens to sequences. Despite these innovations, how to jointly train RL and MTP remains largely unexplored.

Joint MTP-RL Training. Although MTP is a widely used module and optimized during pretraining, its role during RL post-training remains largely unresolved. The RL training frameworks, such as veRL (Sheng et al., 2025) and slime (Zhu et al., 2025), document that backpropagates MTP gradients to the main model causes severe degradation and recommend gradient detachment as the stable default (veRL Team, 2026; Zhao, 2024). Among released models, GLM-5 (GLM-5-Team et al., 2026) explicitly detaches MTP during RL. Nemotron-3 Super (NVIDIA, 2025) adopts a healing approach where MTP is fine-tuned after RL with the main model frozen. And Composer-2 (Research et al., 2026) similarly employs detached cross-entropy training for MTP. Several other models that use MTP (e.g., DeepSeek-V3&V4 (Liu et al., 2024; DeepSeek-AI, 2026), Qwen3-Next (Team, 2025b), StepFun-3.5-Flash (Huang et al., 2026)) have not reported their RL-stage MTP strategy. To our knowledge, no prior work has provided a theoretical explanation for why joint MTP-RL training fails or proposed a principled method to enable their joint training.

3 Joint MTP-RL Training

In this section, we first establish a joint training optimization framework to analyze joint MTP-RL training (§3.1). We then apply this framework to explain the behavior of three MTP regimes (§3.2). Next, we identify the degradation mechanism be-

hind the policy loss regime (§3.3). Finally, we derive Optimal Coefficient Calibration (OCC), an adaptive procedure that calibrates MTP coefficient throughout training (§3.4).

3.1 Theoretical Framework: Effect of MTP on RL Objective

Consider the RL objective of maximizing the expected reward $J(\theta)$. For an L -smooth objective function (Schulman et al., 2015; Zhang et al., 2020), there exists a constant $L > 0$ such that for all θ, θ' :

$$J(\theta') \geq J(\theta) + \langle \nabla J(\theta), \theta' - \theta \rangle - \frac{L}{2} \|\theta' - \theta\|^2, \quad (1)$$

where $\langle \cdot, \cdot \rangle$ stands for dot product. When MTP is introduced, the parameter update yields:

$$\theta_{t+1} = \theta_t + \eta (\nabla J(\theta_t) + \lambda \nabla J_{\text{MTP}}(\theta_t)), \quad (2)$$

where η is the learning rate and λ is the MTP loss coefficient. For notational convenience, we write $g_{\text{RL}} \triangleq \nabla J(\theta_t)$ and $g_{\text{MTP}} \triangleq \nabla J_{\text{MTP}}(\theta_t)$.

Substituting Eq. (2) into inequality (1), we obtain the per-step improvement lower bound:

$$J(\theta_{t+1}) \geq J(\theta_t) + \underbrace{\eta \left(1 - \frac{L\eta}{2}\right) \|g_{\text{RL}}\|^2}_{\text{standard RL improvement}} + \Delta_{\text{MTP}}, \quad (3)$$

where Δ_{MTP} captures the effect of MTP:

$$\begin{aligned} \Delta_{\text{MTP}} = & \underbrace{\eta \lambda (1 - L\eta) \langle g_{\text{RL}}, g_{\text{MTP}} \rangle}_{\text{first-order correlation term}} \\ & - \underbrace{\frac{L\eta^2 \lambda^2}{2} \|g_{\text{MTP}}\|^2}_{\text{second-order penalty term}}. \end{aligned} \quad (4)$$

This decomposition shows that the effect of MTP on $J(\theta)$ is governed by two terms: the first term reflects the directional correlation between RL and MTP gradients, while the second term (always non-positive) represents the per-step perturbation introduced by the MTP gradient. Whether MTP improves or degrades $J(\theta)$ is determined by the relative magnitude of these two terms.

3.2 Analysis of Three Training Regimes

We now apply Eq. (4) to explain the behavior of three MTP training regimes.

Regime 1: Detach. When MTP gradients (usually from cross-entropy) are detached, $g_{\text{MTP}} = \mathbf{0}$ w.r.t the main model, yielding $\Delta_{\text{MTP}} = 0$. MTP does not affect the main model.

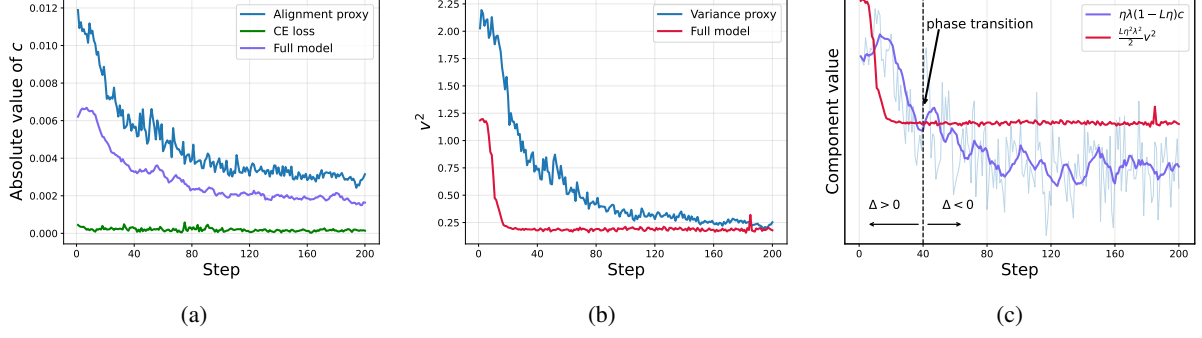


Figure 2: Illustration of the training dynamics of coefficient components. (a) Curves of $|c|$ of full model, CE loss, and alignment proxy. While the correlation of CE loss (green curve) remains near zero, the full model correlation of policy loss declines steadily. The proxy $|\hat{c}_t|$ effectively captures the drift trends. (b) Curves of v^2 of full model and variance proxy. $\hat{v}_t^2$ also captures v^2 and stabilizes around a similar magnitude in the later stage. (c) Curves of first-order correlation term and second-order penalty term. The first-order correlation term decays while the second-order penalty stays persistent. Their crossing flips Δ_{MTP} from positive gain to negative gain.

Regime 2: Cross-Entropy Loss. When MTP cross-entropy loss is backpropagated into the main model, its gradient yields $g_{\text{CE}} = -\nabla \mathcal{L}_{\text{CE}} = \frac{1}{B} \sum_i \nabla_{\theta} \log \pi(y_i | x_i)$ ¹, while the RL gradient is $g_{\text{RL}} = \frac{1}{B} \sum_i A_i \nabla_{\theta} \log \pi(y_i | x_i)$, where A_i denotes the advantage estimate for sample i .

A key property of A_i is *zero-mean*: $\frac{1}{B} \sum_i A_i \approx 0$ (Shao et al., 2024). Under the assumption that A_i are approximately independent of the corresponding per-sample gradient norms $\|\nabla \log \pi_i\|^2$, the expected first-order correlation term satisfies:

$$\mathbb{E}[\langle g_{\text{RL}}, g_{\text{CE}} \rangle] \approx \frac{1}{B^2} \sum_i A_i \|\nabla \log \pi_i\|^2 \approx 0. \quad (5)$$

Since the first-order term is zero in expectation, the second-order penalty dominates:

$$\Delta_{\text{MTP}}^{\text{CE}} \approx -\frac{L\eta^2\lambda^2}{2} \|\nabla \mathcal{L}_{\text{CE}}\|^2 < 0. \quad (6)$$

As shown by the green curve in Figure 2 (a), the correlation of CE loss remains near zero. While in Figure 2 (b), the variance remains substantial. This persistent variance introduces perturbation into $J(\theta)$ and inducing non-negligible performance degradation. This can be intuitively understood by noting that, RL amplifies high-reward samples and suppresses low-reward ones, whereas cross-entropy treats all samples equally regardless of reward values. Consequently, the two objectives conflict in directions and finally hurt training.

Regime 3: Policy Loss. We argue that MTP should adopt the same objective formulation as

¹The sign of $\nabla \mathcal{L}$ is “-” because the RL gradient is an ascent direction on J while CE is descent on $\mathcal{L}$.

the main model (e.g., advantage, trust region, and credit assignment), as $g_{\text{MTP}} = \nabla J_{\text{MTP}}$ points in a direction roughly consistent with the main RL objective. Denoting the gradient inner product as $c \triangleq \langle g_{\text{RL}}, g_{\text{MTP}} \rangle$ and the squared norm of the MTP gradient as $v^2 \triangleq \|g_{\text{MTP}}\|^2$, Eq. (4) yields:

$$\Delta_{\text{MTP}}^{\text{PL}} = \eta\lambda(1-L\eta)c - \frac{L\eta^2\lambda^2}{2}v^2. \quad (7)$$

This is positive when the correlation is larger than the penalty:

$$\begin{aligned} \eta\lambda(1-L\eta)c &> \frac{L\eta^2\lambda^2}{2}v^2 \\ \Rightarrow \lambda &< \frac{2(1-L\eta)c}{L\eta v^2}. \end{aligned} \quad (8)$$

Under this condition, policy loss improves the RL policy improvement lower bound.

3.3 Degradation Behind Policy Loss

Although policy loss is theoretically capable of improving $J(\theta)$, empirical observation in Figure 1 reveals that it initially outperforms the Detach baseline (48.9 vs. 48.3 at 30 step), but later drops below it (38.9 vs. 45.3 at last step). We explain this behavior by analyzing how the two terms in Eq. (7) drift over the course of training.

Early phase: correlation dominance. As the model is far from convergence, gradients are obtained in a steep region of the loss landscape (Li et al., 2018). Both the RL and MTP objectives produce correlated gradients. As shown by the purple curves in Figure 2 (a) and (c), c is large and positive,

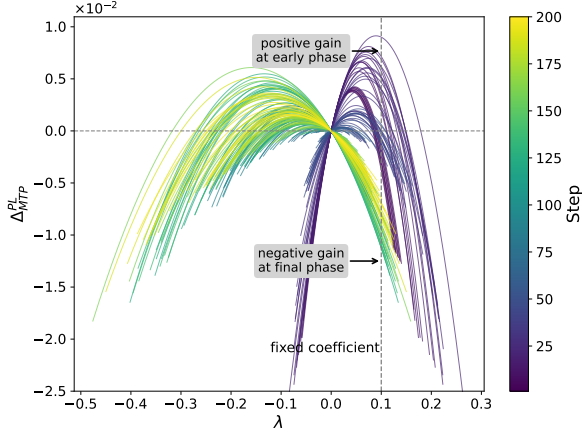


Figure 3: Evolution of the policy-aligned gain across training steps (Eq. (10)). Each curve is a downward-opening parabola in λ , coloured from early (dark purple) to late (bright yellow) steps. The vertical dashed line marks a fixed λ : at early steps, the intersection point of λ and the parabola lies above zero (*positive gain*), but as training progresses the parabola drifts so that the intersection point falls under zero (*negative gain*).

so the correlation term dominates. Consequently, $\Delta_{\text{MTP}} > 0$ and MTP improves RL training².

Late phase: penalty dominance. As the model enters a flat region, the RL and MTP gradients diverge in direction, leading to reduced correlation, while the MTP continues to produce substantial gradient norm (Zhang et al., 2026), so v^2 remains large. As shown in Figure 2 (c), this asymmetry triggers a phase transition:

$$\underbrace{\eta\lambda(1-L\eta)c}_{\text{weakens}} < \underbrace{\frac{L\eta^2\lambda^2}{2}v^2}_{\text{persists}}, \quad (9)$$

causing Δ_{MTP} to flip from positive to negative, and MTP begins to degrade $J(\theta)$.

Figure 2 (c) shows this phase transition. During early training, correlation is strongly positive, indicating that RL and MTP gradients point in similar directions. As training progresses, correlation decays, causing the degradation of $J(\theta)$.

3.4 Optimal Coefficient Calibration (OCC)

To enable an adaptive coefficient calibration that tracks the drift during training, we treat Δ_{MTP} as

²We note that Figure 2 (a) plots $|c|$ rather than c . The raw c is a signed quantity that can be positive or negative. Averaging the signed values produces a curve that hovers near zero. Taking the absolute value preserves the magnitude, making the curve visible.

a downward-opening parabola of λ :

$$\Delta_{\text{MTP}}(\lambda) = \eta(1-L\eta)c \cdot \lambda - \frac{L\eta^2}{2}v^2 \cdot \lambda^2, \quad (10)$$

Figure 3 illustrates the evolutionary trajectory of Eq. (10) across training steps. Under a fixed coefficient, the gain exhibits a drift: during the early phase, the gain remains positive. However, as the training progresses, the gain drifts from positive to negative. This analysis yields a key insight: the MTP coefficient should not be fixed but should adaptively calibrate the drift.

Closed-form optimal coefficient. At every step, we set λ to the optimal point of current parabola, thereby maximizing the per-step improvement. The maximum of Eq. (10) is attained at:

$$\lambda^* = \frac{(1-L\eta)c}{L\eta v^2} = \frac{1-L\eta}{L\eta} \cdot \frac{c}{v^2}. \quad (11)$$

Since the smoothness constant L is unknown in practice, the prefactor $\frac{1-L\eta}{L\eta}$ cannot be computed directly. However, since the constant is fixed during training, it can be absorbed into a global scaling factor. The theoretically meaningful variable that governs the optimal coefficient is the ratio c/v^2 .

Log-probability gradient proxy. Computing full-model gradients is prohibitively expensive in large-scale distributed training systems (Shoeybi et al., 2019). Inspired by OTB (Li et al., 2026) and FIPO (Ma et al., 2026), we use the change in log-probability under small-step updates as a proxy for the gradient:

$$\langle g_1, g_2 \rangle \propto \langle \delta_1, \delta_2 \rangle, \quad \|g\|^2 \propto \|\delta\|^2, \quad (12)$$

where $\delta \triangleq \log \pi_\theta - \log \pi_{\text{old}}$ denotes the log-probability change between current and old policy within small-step updates. This proxy calculation is negligible compared to computing the whole gradient but it serves as a good indicator, as shown by the blue curves in Figure 2. We define two online statistics computed over each training batch:

(1) **Alignment proxy:** $\hat{c}_t = \langle \delta_{\text{RL}}, \delta_{\text{MTP}} \rangle$, estimating the gradient inner product c .

(2) **Variance proxy:** $\hat{v}_t^2 = \|\delta_{\text{MTP}}\|^2$, estimating the auxiliary gradient norm v^2 .

The optimal coefficient per-step is given by:

$$\lambda_t = \lambda_+ \cdot \frac{\hat{c}_t}{\hat{v}_t^2 + \epsilon}, \quad (13)$$

where $\epsilon > 0$ is a small constant to avoid division by zero and λ_+ is a predefined ratio that absorbs the unknown smoothness prefactor.

Table 1: Main results on mathematical reasoning benchmarks across models and algorithms. We report avg@32 accuracy (%). Best results are in **bold**; second best are underlined.

Regime	AIME24	AIME25	AMC	MATH	Minerva	Olympiad	Avg.
<i>MiMo-7B-RL w/ DAPO</i>							
Detach	<u>45.3</u>	<u>36.7</u>	<u>81.9</u>	<u>90.0</u>	<u>39.7</u>	<u>60.1</u>	<u>58.9</u>
CE Loss	16.1	33.3	67.5	83.2	33.8	52.3	47.7
Policy Loss	38.9	<u>36.7</u>	79.5	89.8	<u>39.7</u>	59.7	57.4
OCC (Ours)	45.9	46.7	83.1	91.0	43.4	60.3	61.7
<i>MiMo-7B-RL w/ GSPO</i>							
Detach	43.4	33.3	<u>79.5</u>	<u>90.6</u>	<u>40.1</u>	<u>59.7</u>	<u>57.8</u>
CE Loss	10.3	16.7	63.9	81.8	29.0	44.1	50.0
Policy Loss	37.4	<u>34.9</u>	78.6	88.9	38.4	58.7	56.2
OCC (Ours)	47.6	36.2	84.3	91.8	40.4	60.1	60.1
<i>GLM-4.5-Air w/ DAPO</i>							
Detach	<u>63.8</u>	<u>50.0</u>	<u>84.3</u>	92.8	40.4	<u>63.9</u>	<u>65.9</u>
CE Loss	50.2	33.3	68.7	87.8	40.2	48.6	54.8
Policy Loss	57.3	49.9	83.2	<u>93.1</u>	<u>40.5</u>	62.2	64.4
OCC (Ours)	64.7	52.8	85.4	93.3	42.7	66.6	67.6

4 Experiments

4.1 Experimental Setup

Training data. Building upon the experimental frameworks established in previous attempts (veRL Team, 2026; Zhao, 2024), we utilize DAPO-Math-17k (Yu et al., 2025) as training dataset. This specialized corpus consists of 17k mathematical reasoning prompts, designed to challenge and evaluate the logical depth of LLMs.

RL algorithms. We use DAPO (Yu et al., 2025) as the primary RL algorithm, aligning with the previous attempts from veRL (Sheng et al., 2025) and slime (Zhu et al., 2025). To demonstrate that OCC is algorithm-agnostic, we additionally report results with GSPO (Zheng et al., 2025).

Evaluation benchmarks. We evaluate on multiple widely-adopted mathematical reasoning benchmarks: AIME24 (AoPS, 2024), AIME25 (AoPS, 2025), AMC (Li et al., 2024), MATH (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022) and OlympiadBench (He et al., 2024). Answer correctness is verified using the DeepScaler (Luo et al., 2025) evaluation framework. Results are reported as the average accuracy over 32 runs.

Baseline methods. We use MiMo-7B-RL (Xiomi, 2025) with MTP head as the base model. Also, we additionally use GLM-4.5-Air (GLM-5-Team et al., 2026), a 106B-A12B Mixture-of-

Experts model with MTP head to verify generalization across model scales and architectures. We compare our method with the following regimes from the theoretical framework: (1) **Detach**: MTP heads participate in forward computation but their gradients are detached from the main model. (2) **CE Loss**: MTP backpropagates cross-entropy gradients into the main model. (3) **Policy Loss**: MTP uses the same RL objective as the main model with a fixed coefficient λ . (4) **OCC (Ours)**: dynamically calibrates optimal coefficient according to Eq. (13) with log-probability proxy.

Implementation details. All experiments are conducted using veRL (Sheng et al., 2025) framework. We train for 200 steps with batch size 128 and learning rate $1e-6$. The rollout generation uses a temperature of 1.0. For CE loss, we set $\lambda = 0.1$ following the commonly used auxiliary loss coefficient in prior work. For policy loss, we similarly use $\lambda = 0.1$. For OCC, we set $\lambda_+ = 1.0$ and $\epsilon = 10^{-8}$ as the defaults. All other hyperparameters follow the default configuration in veRL. Training is performed on 128 NVIDIA H20 GPUs.

Setup beyond mathematical reasoning. To test whether OCC is tied to mathematical RLVR, we additionally train under two non-mathematical settings. (1) **Science RLVR**: starting from MiMo-7B-RL, we train with DAPO on a mixture of the SciQ (Welbl et al., 2017) and OpenBookQA (Mi-

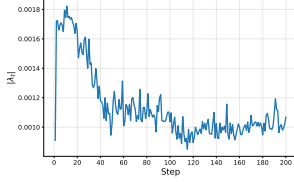


Figure 4: Curve of the absolute value of OCC coefficient λ_t .

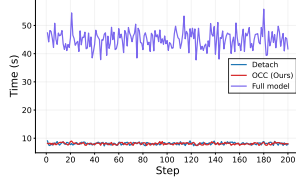


Figure 5: Wall-clock training time of OCC, Detach and Full-model.

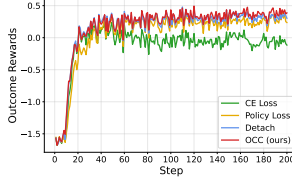


Figure 6: Training reward of CE Loss, Detach, Policy Loss and OCC.

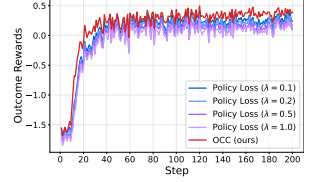


Figure 7: Training reward of the Policy Loss with different coefficient.

haylov et al., 2018) training splits, containing 16,636 multiple-choice science problems. The mixture nearly matches the size of DAPO-Math-17k and keeps the same verifiable-reward structure through exact-match answer checking, while changing the training domain. We evaluate the resulting checkpoints without any target-domain parameter update on GPQA-Diamond (Rein et al., 2023) and ARC-Challenge (Clark et al., 2018) in the 0-shot setting, and on the non-mathematical STEM categories of MMLU-Pro (Wang et al., 2024) in the 5-shot CoT setting; none of these evaluation sets is used for training. (2) **Agentic SWE RL**: we train Qwen3.5-9B (Team, 2026) with GSPO on executable repository-level tasks from SWE-rebench (Badertdinov et al., 2026), where each trajectory interacts with a code repository through an agentic scaffold and receives a verifiable reward from executable tests. We report the resolve rate on all 500 SWE-bench Verified (Jimenez et al., 2024; Chowdhury et al., 2024) instances using the official evaluation harness. Both settings compare Detach, policy loss with $\lambda = 0.1$, and OCC under identical initialization, optimization, and rollout budgets, and reuse $\lambda_+ = 1.0$ selected on MiMo-7B-RL with DAPO without any retuning.

4.2 Main Results

Table 1 presents the results on the benchmarks using MiMo-7B-RL with both DAPO and GSPO. We highlight three findings:

CE loss consistently underperforms Detach.

As shown in Table 1 and Figure 1, across all benchmarks, CE loss falls far below Detach (47.7 vs. 58.9 on average), with the largest gap on AIME24 (-29.2 points). This empirically confirms the theoretical analysis of §3.2: the first-order correlation term vanishes, leaving only the negative second-order penalty. The cross-entropy objective thus injects uncorrelated perturbation into the RL update and systematically suppresses the policy improvement

lower bound.

OCC consistently matches or exceeds Detach.

OCC achieves the best average accuracy (61.7), exceeding Detach by +2.8 points and policy loss by +4.3 points. The largest margin appears on AIME25 (+10.0 points over detach and policy loss), indicating that OCC captures the drift effectively and adjust the coefficient to improve the training. Consistent gains appear MATH (+1.0), AMC (+1.2), and Minerva (+3.7). The same gain is visible in the training-reward trajectories in Figure 6: CE loss degrades below the other regimes. Policy loss peaks then declines, consistent with the rise-then-fall narrative in §3.2. OCC attains the highest reward, showing the capability to improve joint MTP-RL training.

Generalization across RL algorithms. Using GSPO (Zheng et al., 2025) with MiMo-7B-RL, the same result is shown in the middle of Table 1: CE loss remains worst, policy loss underperforms Detach, and OCC is uniformly best, reaching 60.1 average accuracy versus 57.8 for Detach (+2.3). As with DAPO, consistent gains appear across benchmarks, indicating that OCC’s benefit in joint MTP-RL training.

Generalization across base models. Table 1 also evaluates the regimes on GLM-4.5-Air (GLM-5-Team et al., 2026), a 106B-A12B MoE model. The results hold: OCC reaches 67.6 average (vs. 65.9 for Detach, +1.7), with performance gains on every benchmark. This confirms that OCC generalizes across model scales, remaining effective from a dense 7B model up to a 106B MoE.

4.3 Generalization Beyond Mathematical Reasoning

Non-mathematical science RLVR. Table 2 reports the results after training on the science mixture. OCC improves over Detach on all three evaluations, by +1.6 on GPQA-Diamond, +2.0 on

Table 2: Non-mathematical science RLVR: MiMo-7B-RL trained with DAPO on the 16.6K-prompt SciQ + OpenBookQA mixture. GPQA-Diamond and ARC-Challenge are evaluated 0-shot, and the non-mathematical STEM categories of MMLU-Pro 5-shot CoT. We report accuracy (%). Best results are in **bold**.

Regime	GPQA-Diamond	ARC-Challenge	MMLU-Pro	Avg.
Detach	32.5	69.6	61.8	54.6
Policy Loss	29.2	66.3	58.3	51.3
OCC (Ours)	34.1	71.6	63.2	56.3

Table 3: Agentic SWE RL: Qwen3.5-9B trained with GSPO on executable repository-level tasks from SWE-rebench. We report the resolve rate (%) obtained with the official harness on all 500 SWE-bench Verified instances. Best result is in **bold**.

Regime	Detach	Policy Loss	OCC (Ours)
Resolve Rate	58.8	57.2	60.2

ARC-Challenge and +1.4 on MMLU-Pro, raising the macro-average accuracy from 54.6 to 56.3 (+1.7). Policy loss with a fixed coefficient again falls below Detach (51.3 vs. 54.6), repeating the ordering observed on mathematical benchmarks in Table 1. Since λ_+ is transferred unchanged from the mathematical setting, the improvement comes from the online calibration itself rather than from per-domain hyperparameter tuning, indicating that the correlation-drift mechanism of §3.2 is not specific to mathematical rewards.

Agentic SWE RL. Table 3 evaluates the same three regimes in a long-horizon, tool-using setting. Detach resolves 58.8% of the 500 SWE-bench Verified instances, fixed-coefficient policy loss degrades it to 57.2% (−1.6 points), while OCC reaches 60.2% (+1.4 points over Detach). This setting differs from mathematical RLVR in trajectory length, tool use and reward sparsity, yet the ordering of the three regimes is unchanged. Together with the science results, this shows that OCC consistently improves over the detached default across diverse RL objectives, supporting our claim that OCC is task-agnostic.

4.4 Ablation Study

Fidelity of log-probability gradient proxy. The log-probability gradient proxies $\hat{c}_t$ and $\hat{v}_t^2$ can reflect the behaviour of the full model gradient, as fully discussed in previous works (Li et al., 2026; Ma et al., 2026). Figure 2 provides empirical support: the proxy $\hat{c}_t$ in panel (a) exhibits the consis-

tent decay predicted by the analysis for the gradient correlation term, transitioning from a positive correlation-dominant phase to near-zero in the late stage; the variance proxy $\hat{v}_t^2$ in panel (b) remains bounded away from zero and stabilizes around 0.25, mirroring the persistent MTP-gradient variance that drives the second-order penalty.

Figure 8 further validates the fidelity of these proxies by directly comparing them against the full-model gradient quantities: the alignment proxy $\hat{c}_t$ shows strong positive correlation with the true c (left), and the variance proxy $\hat{v}_t^2$ similarly correlates with the full-model second-order penalty v^2 (right). In both panels the linear regression fit confirms that the cheap log-probability proxy is a faithful substitute for the expensive full gradient statistics that drive OCC.

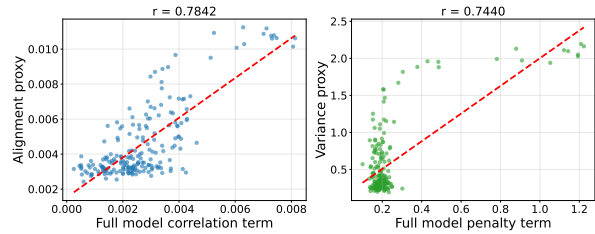


Figure 8: Relationship between the log-probability proxies and the true full-model gradient quantities. Each point corresponds to one training step. **Left:** the alignment proxy plotted against the full-model first-order correlation. **Right:** the variance proxy plotted against the full-model second-order penalty. The proxies exhibit highly positive correlation with the true gradient quantities (r is annotated above).

OCC automatically adjust λ_t . Figure 4 traces the adaptive coefficient $\lambda_t = \lambda_+ \cdot \hat{c}_t / (\hat{v}_t^2 + \epsilon)$ produced by OCC. The trajectory follows the mechanism modeled by Eq. (13) and shown by Figure 2: λ_t is large during the correlation-dominant phase, then decays to near zero once $\hat{c}_t$ drifts. The curve shows that OCC captures the phase transition during training, and effectively adjust λ_t to ensure the positive gain during training.

Training Time Overhead We compare the per-step wall-clock training time under Detach, OCC, and full-model gradient computation. As shown in Figure 5, OCC achieves an average step time of 8.07 s, comparable to Detach at 8.11 s, whereas full-model backpropagation requires 45.06 s, yielding a $5.6\times$ slowdown. This indicates that OCC introduces negligible overhead while avoiding the prohibitive cost of exact MTP gradient compu-

Table 4: Comparison of policy loss with $\lambda \in \{0.1, 0.2, 0.5, 1.0\}$ against Detach and OCC.

Method	AIME24	AIME25	AMC	MATH	Minerva	Olympiad	Avg.
Detach	45.3	36.7	81.9	90.0	39.7	60.1	58.9
Policy Loss ($\lambda=0.1$)	38.9	36.7	79.5	89.8	39.7	59.7	57.4
Policy Loss ($\lambda=0.2$)	38.7	36.6	80.7	89.9	39.5	59.8	57.5
Policy Loss ($\lambda=0.5$)	38.9	36.7	79.5	89.8	39.7	59.7	57.4
Policy Loss ($\lambda=1.0$)	38.8	36.4	78.1	88.3	40.6	57.1	56.6
OCC (Ours)	45.9	46.7	83.1	91.0	43.4	60.3	61.7

tation, thereby enabling efficient adaptive MTP weighting for large-scale distributed RL training.

Policy loss cannot match OCC at any fixed coefficient. An important question is whether the weaker performance of policy loss is due to suboptimal hyperparameter tuning. In Figure 6, we compare the training reward of the three MTP variants with OCC. CE loss shows clear reward degradation, while Detach and policy loss both fall behind OCC in the later stage of training. OCC also achieves the highest reward than other regimes.

Figure 7 and Table 4 sweep the coefficient of policy loss across $\lambda \in \{0.1, 0.2, 0.5, 1.0\}$. Results show that all static coefficient settings are surpassed by OCC on the average and on every individual benchmark, with small λ behaving closely to Detach and large λ collapsing in the late stage. These results show that no fixed coefficient can match OCC. The improvement from OCC therefore comes from its dynamic adjustment w.r.t the drift of gradient alignment and phase transition, rather than from a well-tuned constant coefficient.

Sensitivity of λ_+ . The predefined ratio λ_+ absorbs the smoothness prefactor $\frac{1-L\eta}{L\eta}$ in Eq. (11). Table 5 presents the sensitivity of OCC to different choices of λ_+ . While individual benchmark performance varies, selecting $\lambda_+ = 1.0$ based on the average performance across all benchmarks yields the best overall results. This robustness arises because the adaptive ratio $\hat{c}_t/\hat{v}t^2$ already captures the dominant dynamics, and λ_+ primarily rescales the overall magnitude.

5 Conclusion

We revisit the failure of joint MTP-RL training from an optimization perspective. Starting from an L -smoothness argument, we derived a per-step improvement bound that decomposes the effect of MTP into a first-order correlation term and a second-order penalty. This decomposition uni-

Table 5: Sensitivity analysis of λ_+ . We sweep $\lambda_+ \in \{0.1, 0.2, 0.5, 1.0, 2.0\}$ and compare the performance across various benchmarks. $\lambda_+ = 1.0$ yields the best average performance, indicating it as the preferred choice based on overall results. Best in **bold**.

λ_+	AIME24	AIME25	AMC	MATH	Minerva	Olympiad	Avg.
0.1	47.0	43.8	81.7	90.1	41.5	59.4	60.6
0.2	45.1	44.6	83.3	90.4	42.3	59.7	60.9
0.5	45.6	46.2	82.8	90.7	42.9	60.0	61.4
1.0	45.9	46.7	83.1	91.0	43.4	60.3	61.7
2.0	44.8	45.4	82.0	91.8	42.1	59.6	60.9

fies the Detach, Cross-Entropy, and Policy Loss regimes under a single framework, and exposes a phase transition during training that no static coefficient can track. Guided by this analysis, we proposed **Optimal Coefficient Calibration (OCC)**, which sets the MTP coefficient to its closed-form optimum at every step via a log-probability proxy at negligible cost. Extensive results on multiple benchmarks show that OCC consistently matches or exceeds the detach baseline, demonstrating that MTP can be safely re-introduced into RL post-training and deliver consistent gains.

Acknowledgements

We thank @Mingxuan Yu, @Zhenyu Zhang, and @Qixuan Sun for their support in building and maintaining the code environment, and for their help in diagnosing and resolving implementation issues throughout this project. Their assistance was essential to the large-scale training and evaluation experiments reported in this work.

Limitations

Our theoretical analysis rests on the L -smoothness assumption, which may not hold exactly in non-convex deep learning landscapes; the bound provides qualitative rather than quantitative guidance. The log-probability proxy for gradient alignment is a first-order approximation that may lose fidelity under large learning rates or aggressive policy updates. Our experiments cover mathematical and science RLVR as well as agentic SWE RL, all of which rely on verifiable rewards; generalization to RL post-training with learned reward models or open-ended generation remains to be validated. Finally, the ratio λ_+ absorbs an unknown smoothness-dependent prefactor; while our ablations show robustness across a wide range, the optimal choice may vary across training configurations.

References

- AoPS. 2024. [Aime 2024 dataset](#).
- AoPS. 2025. [Aime 2025 dataset](#).
- Ibragim Badertdinov, Alexander Golubev, Maksim Nekrashevich, Anton Shevtsov, Simon Karasik, Andrei Andriushchenko, Maria Trofimova, Daria Litvinseva, and Boris Yangel. 2026. Swe-rebench: An automated pipeline for task collection and decontaminated evaluation of software engineering agents. *Advances in Neural Information Processing Systems*, 38.
- Yuxuan Cai, Xiaozhuan Liang, Xinghua Wang, Jin Ma, Haijin Liang, Jinwen Luo, Xinyu Zuo, Lisheng Duan, Yuyang Yin, and Xi Chen. 2025. Fastmtp: Accelerating llm inference with enhanced multi-token prediction. *arXiv preprint arXiv:2509.18362*.
- Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.
- Neil Chowdhury, James Aleman, Bowen Baker, Hao Chen, Elizabeth Zheng, Ryan Mostafi, Carroll Wainwright, Patrick Chao, Zaw Aung, et al. 2024. Introducing SWE-bench verified. OpenAI Blog. <https://openai.com/index/introducing-swe-bench-verified/>.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.
- DeepSeek-AI. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence.
- GLM-5-Team, :, Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, Chenzheng Zhu, Congfeng Yin, Cunxiang Wang, Gengzheng Pan, Hao Zeng, Haoke Zhang, Hao-ran Wang, and 168 others. 2026. [Glm-5: from vibe coding to agentic engineering](#). *Preprint*, arXiv:2602.15763.
- Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*.
- Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.
- Ailin Huang, Ang Li, Aobo Kong, Bin Wang, Binxing Jiao, Bo Dong, Bojun Wang, Boyu Chen, Brian Li, Buyun Ma, Chang Su, Changxin Miao, Changyi Wan, Chao Lou, Chen Hu, Chen Xu, Chenfeng Yu, Chengting Feng, Chengyuan Yao, and 196 others. 2026. [Step 3.5 flash: Open frontier-level intelligence with 11b active parameters](#). *Preprint*, arXiv:2602.10604.
- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. Swe-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, volume 2024, pages 54107–54157.
- Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.
- Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857.
- Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31.
- Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. 2024. Numinamath: The largest public dataset in ai4maths

- with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9.
- Yingru Li, Jiawei Xu, Ziniu Li, Jiakai Liu, Wei Liu, Yuxuan Tong, Longtao Zheng, Zhenghai Xue, Yaxiang Zhang, Tianle Cai, et al. 2026. The optimal token baseline: Variance reduction for long-horizon llm-rl. *arXiv preprint arXiv:2602.07078*.
- Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.
- Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025. [Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl](#). Notion Blog.
- Chiyu Ma, Shuo Yang, Kexin Huang, Jinda Lu, Haoming Meng, Shangshang Wang, Bolin Ding, Soroush Vosoughi, Guoyin Wang, and Jingren Zhou. 2026. Fipo: Eliciting deep reasoning with future-kl influenced policy optimization. *arXiv preprint arXiv:2603.19835*.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2381–2391.
- NVIDIA. 2025. [Nvidia nemotron 3: Efficient and open intelligence](#). White Paper.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*.
- Cursor Research, Aaron Chan, Ahmed Shalaby, Alexander Wettig, Aman Sanger, Andrew Zhai, Anurag Ajay, Ashvin Nair, Charlie Snell, Chen Lu, et al. 2026. Composer 2 technical report. *arXiv preprint arXiv:2603.24477*.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.
- Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.
- Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. *Advances in Neural Information Processing Systems*, 31.
- Meituan LongCat Team. 2025a. [Longcat-flash technical report](#). Preprint, arXiv:2509.01322.
- Qwen Team. 2025b. [Qwen3 technical report](#). Preprint, arXiv:2505.09388.
- Qwen Team. 2026. [Qwen3.5: Towards native multi-modal agents](#).
- veRL Team. 2026. Multi-token prediction in verl. <https://verl.readthedocs.io/en/latest/advance/mtp.html>. Accessed: 2026-02-15.
- Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyang Jiang, et al. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.
- Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106.
- LLM-Core-Team Xiaomi. 2025. [Mimo: Unlocking the reasoning potential of language model – from pre-training to posttraining](#). Preprint, arXiv:2505.07608.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.
- Kaiqing Zhang, Alec Koppel, Hao Zhu, and Tamer Basar. 2020. Global convergence of policy gradient methods to (almost) locally optimal policies. *SIAM Journal on Control and Optimization*, 58(6):3586–3612.
- Yaxiang Zhang, Yingru Li, Jiakai Liu, Jiawei Xu, Ziniu Li, Qian Liu, and Haoyuan Li. 2026. Beyond precision: Training-inference mismatch is an optimization problem and simple lr scheduling fixes it. *arXiv preprint arXiv:2602.01826*.

- Chenyang Zhao. 2024. Power up speculative decoding in reinforcement learning. <https://github.com/zhaochenyang20/Awesome-ML-SYS-Tutorial/blob/main/rlhf/slime/spec/readme-en.md>. Part of the Awesome-ML-SYS-Tutorial repository.
- Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. 2025. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*.
- Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. 2024. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583.
- Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. 2025. slime: An llm post-training framework for rl scaling. <https://github.com/THUDM/slime>. GitHub repository. Corresponding author: Xin Lv.

A Detailed Proof of the Policy Improvement Lower Bound

We provide the full derivation of Eq. (3) and Eq. (4) from the main text.

Starting from the L -smoothness condition:

$$J(\theta') \geq J(\theta) + \langle \nabla J(\theta), \theta' - \theta \rangle - \frac{L}{2} \|\theta' - \theta\|^2. \quad (14)$$

This is the standard lower bound for L -smooth functions in the maximization setting, implying Lipschitz continuity of gradients. We further assume $\eta < 1/L$ so that the quadratic approximation remains valid within the update neighborhood.

Setting $\theta = \theta_t$ and $\theta' = \theta_{t+1} = \theta_t + \eta(g_{\text{RL}} + \lambda g_{\text{MTP}})$:

$$J(\theta_{t+1}) \geq J(\theta_t) + \eta \langle g_{\text{RL}}, g_{\text{RL}} + \lambda g_{\text{MTP}} \rangle - \frac{L\eta^2}{2} \|g_{\text{RL}} + \lambda g_{\text{MTP}}\|^2. \quad (15)$$

Expanding the inner product:

$$\langle g_{\text{RL}}, g_{\text{RL}} + \lambda g_{\text{MTP}} \rangle = \|g_{\text{RL}}\|^2 + \lambda \langle g_{\text{RL}}, g_{\text{MTP}} \rangle. \quad (16)$$

Expanding the squared norm:

$$\begin{aligned} \|g_{\text{RL}} + \lambda g_{\text{MTP}}\|^2 &= \|g_{\text{RL}}\|^2 \\ &\quad + 2\lambda \langle g_{\text{RL}}, g_{\text{MTP}} \rangle \\ &\quad + \lambda^2 \|g_{\text{MTP}}\|^2. \end{aligned} \quad (17)$$

Substituting and grouping by terms involving $\|g_{\text{RL}}\|^2$, $\langle g_{\text{RL}}, g_{\text{MTP}} \rangle$, and $\|g_{\text{MTP}}\|^2$:

$$\begin{aligned} J(\theta_{t+1}) &\geq J(\theta_t) + \eta \|g_{\text{RL}}\|^2 + \eta \lambda \langle g_{\text{RL}}, g_{\text{MTP}} \rangle \\ &\quad - \frac{L\eta^2}{2} \|g_{\text{RL}}\|^2 - L\eta^2 \lambda \langle g_{\text{RL}}, g_{\text{MTP}} \rangle \\ &\quad - \frac{L\eta^2 \lambda^2}{2} \|g_{\text{MTP}}\|^2 \\ &= J(\theta_t) + \eta \left(1 - \frac{L\eta}{2}\right) \|g_{\text{RL}}\|^2 \\ &\quad + \eta \lambda (1 - L\eta) \langle g_{\text{RL}}, g_{\text{MTP}} \rangle \\ &\quad - \frac{L\eta^2 \lambda^2}{2} \|g_{\text{MTP}}\|^2. \end{aligned} \quad (18)$$

This yields Eq. (3) with Δ_{MTP} as defined in Eq. (4). Note that under the assumption $\eta < 1/L$, the coefficient $(1 - L\eta) > 0$ and $(1 - L\eta/2) > 1/2 > 0$, ensuring that the standard RL improvement term is strictly positive when $g_{\text{RL}} \neq \mathbf{0}$.

B Vanishing First-Order Term under Cross-Entropy MTP

We provide the full derivation of Eq. (5) and Eq. (6) from the main text. Let $u_i \triangleq \nabla_{\theta} \log \pi(y_i | x_i)$ denote the per-sample log-likelihood gradient. Following the ascent convention of Eq. (2), we write

$$\begin{aligned} g_{\text{RL}} &= \frac{1}{B} \sum_{i=1}^B A_i u_i, \\ g_{\text{MTP}} &= -\nabla \mathcal{L}_{\text{CE}} = \frac{1}{B} \sum_{j=1}^B u_j, \end{aligned} \quad (19)$$

Expanding the first-order correlation term and splitting the diagonal from the cross-sample contributions:

$$\begin{aligned} \langle g_{\text{RL}}, g_{\text{MTP}} \rangle &= \frac{1}{B^2} \sum_{i,j} A_i \langle u_i, u_j \rangle \\ &= \underbrace{\frac{1}{B^2} \sum_i A_i \|u_i\|^2}_{\text{diagonal}} \end{aligned} \quad (20)$$

$$+ \underbrace{\frac{1}{B^2} \sum_{i \neq j} A_i \langle u_i, u_j \rangle}_{\text{cross-sample}}. \quad (21)$$

In a standard RLVR batch, different prompts share little token-level overlap, so the cross-sample inner products $\langle u_i, u_j \rangle$ for $i \neq j$ are small. Summed against the zero-mean advantages, they vanish in expectation:

$$\mathbb{E} \left[\frac{1}{B^2} \sum_{i \neq j} A_i \langle u_i, u_j \rangle \right] \approx 0. \quad (22)$$

Retaining only the diagonal and averaging across batches yields:

$$\mathbb{E}[\langle g_{\text{RL}}, g_{\text{MTP}} \rangle] \approx \frac{1}{B^2} \sum_i A_i \|u_i\|^2. \quad (23)$$

Finally, assuming A_i is approximately independent of $\|u_i\|^2$ — intuitively, A_i is determined by trajectory-level reward while $\|u_i\|^2$ reflects per-token probability structure of the current policy — the expectation factors:

$$\mathbb{E}[A_i \|u_i\|^2] = \mathbb{E}[A_i] \cdot \mathbb{E}[\|u_i\|^2] = 0, \quad (24)$$

since the group-normalised advantage used in GRPO / DAPO / GSPO satisfies $\mathbb{E}[A_i] = 0$ by construction.

Substituting this vanishing first-order term into Eq. (4) recovers Eq. (6):

$$\mathbb{E}[\Delta_{\text{MTP}}^{\text{CE}}] \approx -\frac{L\eta^2\lambda^2}{2} \mathbb{E}[\|\nabla\mathcal{L}_{\text{CE}}\|^2] < 0, \quad (25)$$

which is strictly negative whenever $\nabla\mathcal{L}_{\text{CE}} \neq \mathbf{0}$. CE loss therefore injects pure perturbation into the RL update without a correlation gain, and the magnitude of the damage scales quadratically with λ . This is consistent with the flat, near-zero $|c|$ curve of CE loss in Figure 2 (a) and the large benchmark gap reported for CE loss in Table 1.

C Derivation of the Optimal Adaptive Weight

Treating Δ_{MTP} as a function of λ :

$$f(\lambda) = \eta(1 - L\eta) c \cdot \lambda - \frac{L\eta^2}{2} v^2 \cdot \lambda^2, \quad (26)$$

where $c = \langle g_{\text{RL}}, g_{\text{MTP}} \rangle$ and $v^2 = \|g_{\text{MTP}}\|^2$. This is a concave quadratic in λ (since $-\frac{L\eta^2}{2}v^2 < 0$). Setting $f'(\lambda) = 0$:

$$\begin{aligned} \eta(1 - L\eta) c - L\eta^2 v^2 \lambda &= 0 \\ \implies \lambda^* &= \frac{(1 - L\eta) c}{L\eta v^2}. \end{aligned} \quad (27)$$

The maximum value of the perturbation at this optimum is:

$$f(\lambda^*) = \frac{\eta(1 - L\eta)^2 c^2}{2L\eta v^2} = \frac{(1 - L\eta)^2 c^2}{2Lv^2} > 0, \quad (28)$$

D Experiment Details

Experiments are implemented in `veRL` (Sheng et al., 2025) with the Megatron-LM training backend (Shoeybi et al., 2019) and SGLang rollout engine (Zheng et al., 2024). The four regimes (Detach, CE loss, policy loss, OCC) share the same base configuration and differ only in (i) whether MTP gradients flow into the main model and (ii) the choice of MTP loss coefficient. GSPO runs replace the DAPO advantage estimator and surrogate loss while keeping everything else unchanged.

Dataset and benchmark licenses. We use publicly released training and evaluation resources and follow the license metadata declared by their corresponding dataset or repository pages. The DAPO-Math-17k training dataset is released under Apache-2.0. For evaluation benchmarks,

Table 6: Effect of clipping λ_t to non-negative values. **OCC-Clip** clips the ratio to $\max(0, \hat{c}_t)/\hat{v}_t^2$; **OCC-NoClip** keeps the signed ratio. Final avg@32 accuracy (%) on MiMo-7B-RL + DAPO. Best in **bold**.

Variant	AIME24	AIME25	AMC	MATH	Minerva	Olympiad	Avg.
OCC-Clip	43.9	44.1	81.8	90.3	42.1	59.6	60.3
OCC-NoClip (Ours)	45.9	46.7	83.1	91.0	43.4	60.3	61.7

AIME24 and AIME25 are released under Apache-2.0, MATH/MATH-500 and Minerva Math are released under MIT, and the official OlympiadBench dataset release is listed as Apache-2.0 while its accompanying official evaluation-code repository is MIT-licensed. The public AMC23 benchmark release used in our evaluation does not declare an explicit license on its current dataset card. The DeepScaleR evaluation framework is MIT-licensed.

For evaluation, we use the DeepScaler (Luo et al., 2025) evaluation framework with 32 independent samples per prompt (avg@32), using temperature 1.0 and top- $p = 0.7$ at validation, matching the rollout configuration. All reported numbers are averages over 32 decodes; random seeds for data shuffling and sampling follow the `veRL` defaults.

E Clipping the Adaptive Coefficient

A natural question is whether the online proxy $\lambda_t = \lambda_+ \cdot \hat{c}_t/(\hat{v}_t^2 + \epsilon)$ should be constrained to be positive $\max(0, \hat{c}_t)/\hat{v}_t^2$, that is, whenever the alignment proxy $\hat{c}_t$ becomes negative, should we (i) pass the negative value through (**OCC-NoClip**, our default), or (ii) clip it to zero (**OCC-Clip**)?

Table 6 compares the two variants on MiMo-7B-RL + DAPO. OCC-NoClip outperforms OCC-Clip by +1.3 average points, with the largest gap on AIME25 (+2.6). We interpret this as follows. The optimal MTP coefficient at each step is proportional to the signed ratio $\lambda_t \propto \hat{c}_t/\hat{v}_t^2$, where $\hat{c}_t$ captures the alignment between RL and MTP gradients.

Crucially, the closed-form optimum is symmetric in the sign of c : the maximum value of the parabola $\Delta_{\text{MTP}}(\lambda)$ is $(1 - L\eta)^2 c^2 / (2Lv^2) \geq 0$ regardless of whether c is positive or negative, and is attained at a λ^* that shares the sign of c . When $c < 0$, forcing $\lambda = 0$ collapses the gain to zero, which is strictly worse than the negative- λ^* optimum; clipping therefore replaces the maximizer with a strictly suboptimal projection on half of the parameter axis. Geometrically, a negative λ flips the MTP gradient before it is added to the RL direction, so when the unflipped MTP gradient points

against RL (i.e., $c < 0$), the flipped version is by construction aligned with RL — negative λ is precisely the move that converts a hostile auxiliary signal into a corrective one. Clipping $\hat{c}_t$ to positive values (OCC-Clip) ignores these instances, preventing the model from applying corrective updates that could counteract harmful MTP directions. In contrast, keeping the signed ratio (OCC-NoClip) allows both positive and negative contributions to λ_t , enabling the model to dynamically adjust the MTP coefficient to maximize per-step improvement. This leads to more effective calibration throughout training, yielding consistently higher performance across tasks.

F Potential Risks

Our work aims to improve the efficiency and stability of post-training large language models through joint MTP-RL optimization. While the method is designed to enhance reasoning, such improvements may also affect how models are used in broader settings. Since RLVR depends on verifiable rewards, models may become sensitive to the design and quality of the reward signal, making robustness checks important.

G Use of AI Assistants

AI assistants were used solely to support the writing and editing process of this paper. Specifically, they were employed to polish the language, improve clarity and readability, and correct grammatical errors. All technical ideas, theoretical analysis, method design, experimental setup, results, and conclusions were developed and verified by the authors. The authors carefully reviewed and edited all AI-assisted text to ensure correctness, consistency, and alignment with the paper’s intended meaning.