MODULATING LANGUAGE MODEL REINFORCEMENT LEARNING WITH INTERNAL DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) has become a key approach for improving large language models (LLMs), using feedback from human preferences, verifiable rewards, model signals, or environment interactions. Existing methods mainly determine whether a behavior should be reinforced or suppressed, but largely ignore how strongly the model should learn from it. We find that token representations differ substantially in how they evolve across layers: some stabilize early, while others continue to change until the final layers. These internal dynamics do not indicate whether a decision is good or bad; instead, they provide a natural signal for modulating the strength of policy updates. Based on this observation, we introduce **Internal Dynamics Modulation (IDM)**, a lightweight operator that uses cross-layer representation dynamics to modulate the magnitude of existing policy advantages while preserving their sign. IDM is *lightweight* and *sign-preserving*, and can be inserted after an existing advantage estimator without changing the underlying reward source or policy optimization framework. Across different RL scenarios, IDM consistently improves the corresponding baselines, with larger gains on harder and longer-horizon settings. At larger model scales, internal dynamics also identify critical positions more reliably than entropy or log-probability. These results suggest that internal dynamics provide a general modulation signal complementary to the source of reinforcement feedback, offering a simple way to modulate the strength of policy updates without changing their sign.

1 INTRODUCTION

Reinforcement learning (RL) has become a central approach for improving large language models (LLMs). Early work mainly relied on human preference feedback (Ouyang et al., 2022), while more recent methods use verifiable outcomes (Shao et al., 2024; DeepSeek-AI et al., 2025), model-based or internal signals (Zhao et al., 2026), self-generated feedback for test-time adaptation (Zuo et al., 2025; Wu et al., 2025b), and direct interaction with environments (Wang et al., 2025; 2026). Despite these different feedback sources, they usually follow the same basic pipeline: feedback is converted into an advantage, and the advantage is then used to update the policy. The sign of the advantage determines whether a sampled behavior should become more or less likely, while its magnitude determines how strongly it affects the update. Existing methods improve this signal through better rewards, verifiers, process supervision, and advantage estimators (Lightman et al., 2023; Wang et al., 2024a). However, the magnitude of the update is still mainly determined by reward values, returns, value estimates, or group statistics, with little consideration of how the model itself formed the decision.

This leaves a potentially useful source of information unused. Consider two positions in the same successful trajectory. One may be a routine continuation whose representation becomes stable after only a few layers, while another may correspond to a more involved decision whose representation continues to change until the final layers. Outcome-level RL may assign both positions the same advantage, and even process-level methods determine update strength from external or learned feedback. Neither directly uses the model's own representation dynamics. Since token representations evolve as they pass through the network, these cross-layer changes provide a natural model-internal signal. Some token states approach their final representation early, while others continue to change substantially across later layers (Belrose et al., 2023; Wu et al., 2025a; Chen et al., 2026).

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.

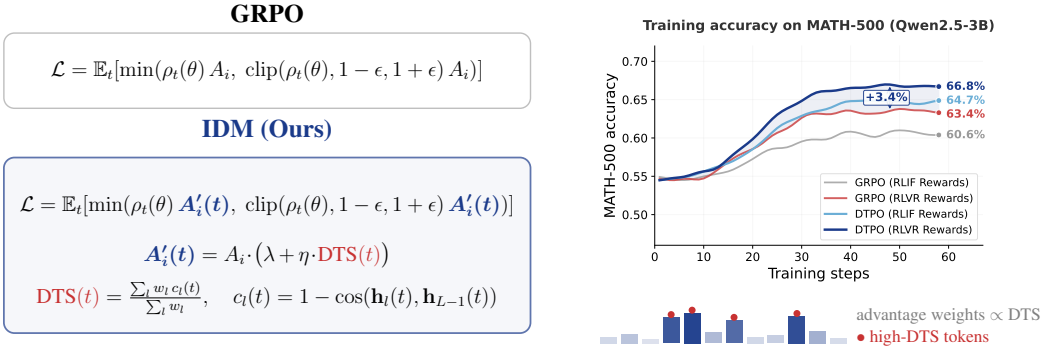

Figure 1: **IDM vs. GRPO.** *Left*: Standard GRPO applies the same sequence-level advantage A_i to all response tokens. IDM modulates this advantage at either sequence or token granularity using DTS. *Right*: Representative MATH-500 training curves from the core RLVR and RLIF experiments.

We refer to this behavior as *internal dynamics*. It is different from output entropy or token log-probability: those signals describe the final predictive distribution, whereas internal dynamics describe how the representation evolves before the prediction is produced. Importantly, internal dynamics are not a reward signal. Large cross-layer changes do not imply that a token is correct, and early stabilization does not imply that it is wrong. Instead, once the base advantage determines whether a behavior should be reinforced or suppressed, internal dynamics can be used to modulate how strongly that behavior contributes to the update. This leads to a simple question:

> *Can internal dynamics modulate how strongly a language model learns from a behavior, without changing whether that behavior is reinforced or suppressed?*

We answer this question with **Internal Dynamics Modulation (IDM)**, a lightweight operator that is applied after an existing advantage estimator. IDM first measures how the hidden representation at each generated position evolves toward its final-layer representation. It then maps this signal to positive modulation weights and multiplies the original advantages by these weights.

IDM uses positive modulation weights, so a positive advantage remains positive and a negative advantage remains negative. This sign-preserving property ensures that internal dynamics modulate how strongly the policy learns from a behavior, while the underlying feedback still determines whether that behavior is reinforced or suppressed. IDM does not replace the underlying reward, advantage estimator, or policy-optimization framework. Instead, it acts as an additional modulation layer between advantage estimation and policy optimization, making the same interface applicable to different feedback sources and levels of granularity.

We first evaluate IDM in RLVR and RLIF, where it consistently improves the corresponding baselines across the reported reasoning, code, language-understanding, and instruction-following benchmarks. We then extend the analysis in two directions. First, we examine whether internal dynamics capture behaviorally sensitive intermediate states beyond final-layer uncertainty. Second, we apply the same modulation interface to self-generated test-time rewards and long-horizon environment feedback. These experiments show that IDM can operate across distinct feedback regimes while retaining the same basic advantage-modulation mechanism. Our contributions are summarized as follows:

- We separate two roles of policy advantages: determining whether a behavior should be reinforced or suppressed, and determining how strongly the policy should learn from it. We identify cross-layer representation dynamics as a model-internal signal for the second role.
- We introduce Internal Dynamics Modulation (IDM), a lightweight and sign-preserving operator that reweights existing advantages at either sequence or token granularity while retaining the original reward source and policy-optimization framework.
- We evaluate IDM across different RL scenarios. IDM consistently improves the corresponding baselines, provides larger gains on harder and longer-horizon settings, and identifies critical positions more reliably than entropy or log-probability at larger model scales.

2 Related Work

2.1 Reinforcement Learning for Large Language Model Reasoning

Reinforcement learning (RL) has become instrumental in refining large language models (LLMs). Early pivotal work centered on reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; Rafailov et al., 2023), while more recent reasoning focused methods increasingly rely on reinforcement learning with verifiable rewards (RLVR), where supervision comes from automatically checkable outcomes for tasks like mathematical reasoning and code generation (OpenAI, 2024; DeepSeek-AI et al., 2025; Hu et al., 2025). While RLVR is largely confined to settings with verifiable gold solutions or exhaustive test cases, reinforcement learning from internal feedback (RLIF) replaces external rewards with intrinsic signals such as entropy or self-certainty (Zhang et al., 2025; Zhao et al., 2026). Robust policy optimization algorithms such as GRPO (Shao et al., 2024) and its variants (Yu et al., 2025; Liu et al., 2025) have further improved the effectiveness of RLVR and RLIF. Nevertheless, both RL paradigms emphasize outcome-based rewards, which hinders generalization to tasks requiring nuanced, process-level feedback.

A major direction beyond outcome supervision is process supervision (Lightman et al., 2023), which evaluates intermediate reasoning steps rather than only the final outcome. Recent works on process reward models (PRMs) and step-level verifiers (Wang et al., 2024a; Luo et al., 2025) show that process supervision can outperform outcome supervision on mathematical reasoning. These methods provide denser guidance for long-horizon reasoning, but also typically rely on explicit step annotations or auxiliary verifiers to score reasoning traces (Setlur et al., 2024; Khalifa et al., 2025). Despite these advances, both outcome-based and process-based RL signals remain centered on correctness, focusing on whether the final answer or intermediate steps are valid. However, correctness alone fails to capture how reasoning is internally distributed, overlooking an orthogonal dimension that reflects the depth and allocation of computation across tokens (Gupta et al., 2025; Csordás et al., 2025). This dimension offers a different lens for supervising reasoning, beyond correctness-based signals.

2.2 Internal Signals and Test-Time Scaling

Another relevant line of work studies how to identify better reasoning traces at inference time. Self-consistency (Wang et al., 2023) and related test-time scaling methods (Yao et al., 2023) improve performance by sampling multiple chains-of-thought (CoT) (Wei et al., 2022) and selecting among them. Recent work further suggests that reasoning quality is reflected not only in output length or final confidence, but also in internal computation patterns. The tuned lens shows that model predictions are progressively refined across layers (Belrose et al., 2023), and later work uses cross-layer uncertainty signals to improve generation quality and factuality (Wu et al., 2025a). Most closely related to our motivation, Chen et al. (2026) show that so-called deep thinking tokens, whose predictions undergo substantial revisions across layers, correlate strongly with reasoning correctness. IDM builds on this emerging perspective but differs in usage: prior work mainly exploits such signals for probing, decoding, or reranking at test time, whereas IDM incorporates internal dynamics into policy optimization as modulation signals.

3 Preliminary Analysis of Internal Dynamics

3.1 Depth-Transition Score (DTS)

Our approach is motivated by the observation that token representations can evolve differently across model depth (Chen et al., 2026). Some representations approach their final-layer state early, whereas others continue to undergo substantial cross-layer changes in later layers. We summarize this behavior through the **Depth-Transition Score (DTS)**, a continuous token-level measure of internal representation dynamics.

Per-Layer Signal. We consider an autoregressive language model π_θ with L transformer layers (Vaswani et al., 2017) and hidden dimension d. Given a generated completion $y_i = (t_1, t_2, \ldots, t_T)$, we perform an additional forward pass to extract intermediate representations from

all L transformer layers. For each token t, this yields a sequence of hidden states:

$$\mathcal{H}(t) = \{\mathbf{h}_0(t), \mathbf{h}_1(t), \ldots, \mathbf{h}_{L-1}(t)\}, \quad \mathbf{h}_l(t) \in \mathbb{R}^d, \tag{1}$$

where $\mathbf{h}_l(t)$ denotes the residual stream representation at layer l. To quantify how strongly the representation changes across model depth, we measure how its representation evolves toward the final layer. Specifically, we compute the cosine distance between each intermediate layer representation and the final-layer representation:

$$c_l(t) = 1 - \cos\big(\mathbf{h}_l(t), \mathbf{h}_{L-1}(t)\big) = 1 - \frac{\mathbf{h}_l(t) \cdot \mathbf{h}_{L-1}(t)}{\|\mathbf{h}_l(t)\| \, \|\mathbf{h}_{L-1}(t)\|}. \tag{2}$$

The value $c_l(t) \in [0, 2]$ captures the magnitude of representation change: values near zero indicate that the intermediate representation is already close to its final-layer state, while larger values indicate a larger cross-layer transition.

Layer-Wise Weighting. Not all layers contribute equally to representation refinement (Tenney et al., 2019; Geva et al., 2021). Early layers are dominated by representation formation, whereas later layers are closer to the model's final prediction. Our empirical analysis further shows that reasoning-relevant token types exhibit stronger representation changes in later layers. We therefore assign larger weights to deeper layers:

$$w_l = \exp\left(\gamma \cdot \frac{l}{L-2}\right), \quad \gamma \geq 0, \tag{3}$$

where L is the total number of layers and γ controls the degree of emphasis on deeper layers. When $\gamma = 0$, the weighting reduces to a uniform scheme. As γ increases, deeper layers receive progressively higher weights. This design encourages the depth signal to focus on layers where meaningful reasoning-related computation is more likely to occur.

Depth-Transition Score. Building on the per-layer depth signal $c_l(t)$ and the layer-wise weights w_l, we aggregate these signals into a single token-level measure. Specifically, we define the Depth-Transition Score (DTS) for each token t as a weighted average over selected intermediate layers:

$$\mathrm{DTS}(t) = \frac{\sum_{l \in \mathcal{S}} w_l \cdot c_l(t)}{\sum_{l \in \mathcal{S}} w_l}, \tag{4}$$

where $\mathcal{S}$ denotes the set of layer indices. In this work, $\mathcal{S}$ include all layers $\{0, 1, \ldots, L-2\}$, because we find a subset of layers delivers suboptimal results. This formulation combines local representation changes across layers while placing greater emphasis on deeper layers. As a result, $\mathrm{DTS}(t)$ provides a normalized measure of the token's internal representation dynamics across model depth.

Relation to Prior Measures. Chen et al. (2026) define deep thinking tokens explicitly via discrete prediction revision through the logit lens via projecting each layer's hidden state to vocabulary space and compute the Jensen–Shannon divergence (JSD) (Lin, 1991) between the intermediate-layer distribution and the final-layer distribution. Our DTS differs in three ways: (1) it operates directly in hidden state space without vocabulary projection, avoiding the noise of uncalibrated intermediate logits; (2) it produces a continuous score rather than a binary indicator; and (3) it is computationally cheaper, requiring only cosine similarity rather than matrix multiplication with the vocabulary head. Accordingly, we use DTS as a continuous internal-dynamics signal for policy modulation rather than as a binary indicator of reasoning depth.

3.2 DTS Signal Properties

Before integrating DTS into RL training, we first verify whether it provides meaningful signals for reasoning analysis. Concretely, we conduct experiments using Qwen2.5-1.5B and Qwen2.5-3B (Qwen Team et al., 2024) on 100 randomly sampled questions from GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021), respectively. For each question, we sample model responses and compute token-level DTS on these outputs.

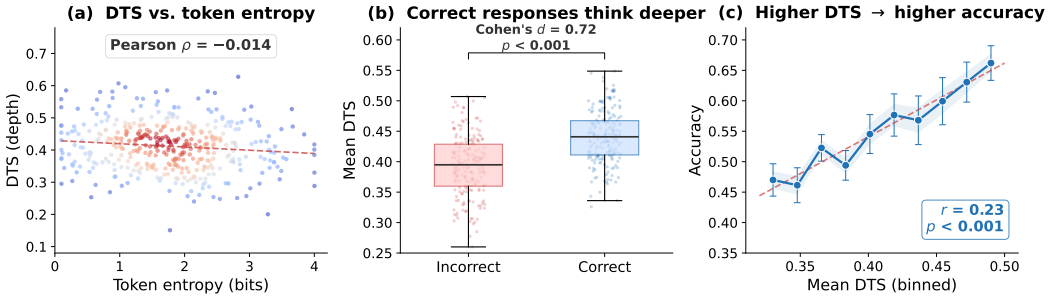

Figure 2: **DTS signal properties.** (a) DTS is weakly correlated with token entropy, capturing complementary information. (b) Correct responses exhibit higher mean DTS. (c) Response accuracy increases with binned mean DTS.

DTS Is Complementary to Entropy. For each generated token, we compute both DTS (Eq. 4) and token-level entropy $H(t) = -\sum_v p_v \log p_v$ from the output logits. Across all tokens, we obtain $\rho(\text{DTS}(t), H(t)) = -0.014 \pm 0.103$. The near-zero Pearson correlation indicates that DTS captures a complementary signal from entropy. A token can have *low entropy* (the model is confident in its output) yet *high DTS* (the layers debated intensely before convergence). This commonly occurs for mathematical operators at critical computation steps: the model is confident about outputting "=", while its internal representation still undergoes substantial revision before convergence. Additional continuation-sensitivity and test-time aggregation analyses in Appendix A.2 further examine how DTS differs from output uncertainty at larger model scales.

DTS Correlates with Response Correctness. We verify that DTS carries a meaningful learning signal by examining its relationship with response correctness. For each response, we compute the mean DTS across all tokens and compare the distributions between correct and incorrect responses. As shown in Figure 2(b), correct responses exhibit significantly higher mean DTS (Cohen's $d = 0.72$, $p < 0.001$), suggesting that deeper internal computation is associated with successful reasoning. Figure 2(c) further shows a monotonic positive relationship between DTS and accuracy when responses are binned by their mean DTS ($r = 0.23$, $p < 0.001$). This provides direct empirical motivation for using DTS as an auxiliary modulation signal in RL.

High-DTS Tokens Are Enriched for Reasoning-Relevant Operators. We further inspect which types of tokens receive high DTS. We categorize generated tokens into four semantic types: OPERATOR ($=, +, -, \times, /$), NUMBER (digits), TEXT (ordinary words), and CONNECTIVE (e.g., "therefore", "so"). We then compare the token-type distribution of all generated tokens with that of the highest-DTS tokens, as summarized in Table 1.

Table 1: Token type distribution for Qwen2.5-1.5B on GSM8K: all tokens vs. high-DTS tokens.

	OPERATOR	NUMBER	TEXT	CONNECTIVE
All	11.1%	35.7%	51.1%	2.1%
High-DTS	**65.0%**	5.0%	25.0%	5.0%
Ratio	**5.9**×	0.14×	0.49×	2.4×

High-DTS regions are strongly enriched for OPERATOR tokens: their proportion increases from 11.1% among all tokens to 65.0% among high-DTS tokens, corresponding to a 5.9× enrichment. This indicates that DTS tends to highlight tokens associated with actual symbolic or arithmetic transitions. In contrast, NUMBER and ordinary TEXT tokens are under-represented among high-DTS tokens. Many numbers are copied directly from the problem statement or previous steps, and many text tokens belong to common reasoning templates; both cases require less cross-layer refinement.

Figure 3 further shows that the DTS signal for reasoning-relevant tokens is concentrated in deeper layers. In particular, OPERATOR tokens exhibit larger cosine distances in the final portion of the network, suggesting that late layers contribute more strongly to reasoning-related representation refinement. This observation motivates the exponential layer weighting used in DTS, where deeper layers receive larger weights.

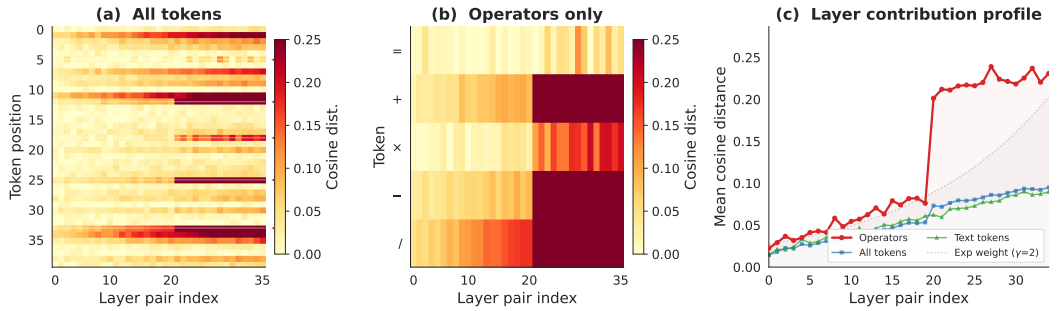

Figure 3: Layer-wise analysis of DTS for Qwen2.5-3B on MATH. Operator tokens exhibit larger cosine-distance transitions in later layers than the average token, motivating the deeper-layer weighting used in DTS.

4 Internal Dynamics Modulation (IDM)

4.1 Preliminary

We build on Group Relative Policy Optimization (GRPO) (Shao et al., 2024) as a representative policy-optimization framework. For each prompt, GRPO samples a group of responses $\{y_i\}_{i=1}^{G}$ and computes a scalar advantage $A_i = (R_i - \mu_g)/(\sigma_g + \epsilon_g)$, where R_i denotes the reward and μ_g, σ_g are the group reward statistics. The policy is optimized with the clipped objective

$$\mathcal{L}_{\text{GRPO}} = \mathbb{E}_t \left[\min \left(\frac{\pi_\theta(t|x)}{\pi_{\theta_{\text{old}}}(t|x)} A_i,\ \text{clip} \left(\frac{\pi_\theta(t|x)}{\pi_{\theta_{\text{old}}}(t|x)}, 1-\epsilon, 1+\epsilon \right) A_i \right) \right] - \beta\, \text{KL}(\pi_\theta \| \pi_{\text{ref}}). \quad (5)$$

Standard GRPO applies the same response-level advantage to every generated token. IDM instead uses internal representation dynamics to modulate this learning signal at either response or token granularity.

4.2 IDM for Policy Optimization

We map DTS to a positive modulation weight

$$f(\text{DTS}) = \lambda + \eta \cdot \text{DTS}, \quad (6)$$

where $\lambda > 0$ provides a base learning signal and $\eta \geq 0$ controls the modulation strength. Since DTS is non-negative, the resulting weight preserves the sign of the original advantage.

For response $y_i = (t_1, \ldots, t_T)$, we first define its average internal-dynamics score as

$$\overline{\text{DTS}}_i = \frac{1}{T} \sum_{t=1}^{T} \text{DTS}(t). \quad (7)$$

IDM then constructs the modulated advantage as

$$A_i'(t) = \begin{cases} A_i\, f(\overline{\text{DTS}}_i), & \text{IDM-Seq}, \\ A_i\, f(\text{DTS}(t)), & \text{IDM-Tok}. \end{cases} \quad (8)$$

IDM-Seq applies a single response-level modulation weight to all tokens, providing a coarse but stable estimate of overall internal dynamics. IDM-Tok instead assigns token-specific weights, enabling finer-grained modulation while potentially introducing greater token-level variance. Both variants retain the original GRPO objective in Eq. (5), replacing the base advantage A_i with the modulated advantage $A_i'(t)$.

In the additional online RL experiments, where intermediate hidden states are not exposed by the rollout engine, we use token surprisal as a lightweight modulation signal while retaining the same advantage-reweighting interface.

Table 2: Main results: IDM on RLVR (reward-verified RL). ↑ higher is better. **Bold**: best within each model scale. Shaded : best within each column per model scale.

		Mathematical		*Code*		*Language*	*Instruction*
Model	Method	MATH ↑	GSM8K ↑	LCB ↑	CRUX ↑	MMLU-Pro ↑	AlpacaEval ↑
Qwen2.5-3B	Base	54.4	67.3	9.3	23.6	37.7	3.72
	+ GRPO	63.4	80.9	8.5	34.1	40.3	6.91
	+ IDM-Seq	**66.8** +3.4	83.3 +2.4	**10.9** +2.4	35.8 +1.7	**41.9** +1.6	**7.24** +0.33
	+ IDM-Tok	66.6 +3.2	**83.7** +2.8	10.6 +2.1	**36.3** +2.2	41.7 +1.4	7.18 +0.27
Qwen2.5-7B	Base	55.3	63.6	2.6	17.8	49.7	4.46
	+ GRPO	75.2	82.6	20.0	53.8	51.1	8.52
	+ IDM-Seq	**79.7** +4.5	**87.7** +5.1	**21.3** +1.3	**55.1** +1.3	**52.8** +1.7	**9.14** +0.62
	+ IDM-Tok	78.3 +3.1	87.1 +4.5	20.8 +0.8	54.7 +0.9	52.5 +1.4	8.97 +0.45

Table 3: Main results: IDM on RLIF (reward-free self-certainty RL). ↑ higher is better. **Bold**: best within each model scale. Shaded : best within each column per model scale.

		Mathematical		*Code*		*Language*	*Instruction*
Model	Method	MATH ↑	GSM8K ↑	LCB ↑	CRUX ↑	MMLU-Pro ↑	AlpacaEval ↑
Qwen2.5-3B	Base	54.4	67.3	9.3	23.6	37.7	3.72
	+ Intuitor	60.6	78.3	15.3	41.6	37.9	7.10
	+ IDM-Seq	**64.7** +4.1	81.3 +3.0	**18.1** +2.8	**43.2** +1.6	**40.6** +2.7	**7.53** +0.43
	+ IDM-Tok	64.6 +4.0	**81.8** +3.5	17.8 +2.5	42.8 +1.2	40.3 +2.4	7.41 +0.31
Qwen2.5-7B	Base	55.3	63.6	2.6	17.8	49.7	4.46
	+ Intuitor	75.0	87.3	19.0	57.4	51.4	12.76
	+ IDM-Seq	**80.0** +5.0	**88.6** +1.3	**22.4** +3.4	**59.1** +1.7	**53.7** +2.3	**13.02** +0.26
	+ IDM-Tok	77.7 +2.7	88.5 +1.2	22.2 +3.2	58.8 +1.4	53.5 +2.1	12.91 +0.15

5 EXPERIMENTS

5.1 SETUP

Model. We evaluate the algorithm's performance on mathematical reasoning, code reasoning, language understanding and instruction following tasks. We use Qwen2.5-3B and Qwen2.5-7B (Qwen Team et al., 2024) as policy models. For baselines, we utilize GRPO (Shao et al., 2024) as the RLVR baseline and Intuitor (Zhao et al., 2026) as the RLIF baseline. We instantiate IDM on both GRPO and Intuitor by shaping their advantages, considering both sequence-level and token-level variants.

Training. GRPO, Intuitor and their IDM variants are trained for 1 epoch with 8×H200 GPUs on the training split of the MATH dataset (Hendrycks et al., 2021), which contains 7,500 problems. For a fair comparison, all algorithms share identical hyperparameters without additional tuning. We use the GRPO objective with group-relative advantage normalization across $N = 8$ generations per prompt for Qwen2.5-3B or $N = 4$ generations for Qwen2.5-7B. For Qwen2.5-3B, the learning rate is $3e-6$ and the KL penalty β is 0.005. For Qwen2.5-7B, the learning rate is $1e-6$ and the KL penalty β is 0.01. Learning rate follows a cosine schedule with 10% warmup, and the effective batch size is set to 128. For IDM, the only additional cost is one forward pass with `output_hidden_states=True` per training step to compute the DTS. This adds ∼20% wall-clock overhead (46s→55s per step for 3B). No additional parameters are introduced. We utilize all $L-1$ pairs in layer selection, and set $\gamma = 2.0$. For shaping function, we set $\lambda = 0.5$ and $\eta = 0.5$. For the additional online TTRL and agentic experiments, we use token surprisal as the modulation signal and normalize token-level weights to unit mean within each response.

Evaluation. We evaluate performance on the following benchmarks: (1) *Mathematical Reasoning*: MATH500 and GSM8K, using the `lighteval` library (Habib et al., 2023). (2) *Code Reasoning*: CRUXEval-O (Gu et al., 2024), using the `ZeroEval` framework (Lin, 2024), and LiveCodeBench v6 (LCB) (Jain et al., 2024). (3) *Language understanding*: MMLU-Pro (Wang et al., 2024b), an enhanced benchmark designed to evaluate language understanding models across broader and more challenging tasks. (4) *Instruction following*: AlpacaEval 2.0 (Dubois et al., 2024) with length-controlled win rates, judged by GPT-4.1 (OpenAI, 2025). Evaluations use the same chat-style prompting format as

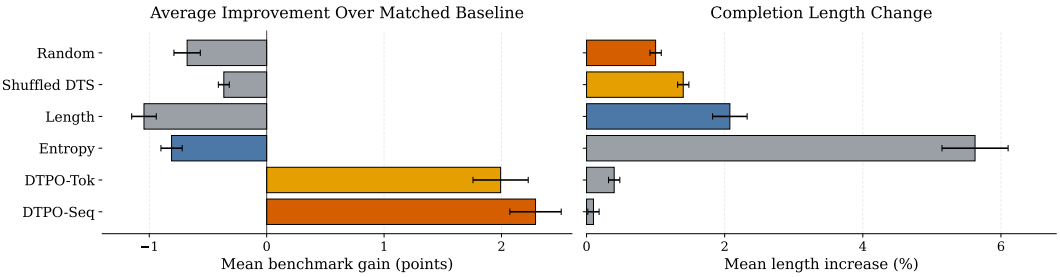

Figure 4: Matched shaping controls test whether IDM gains stem from generic reweighting. *Left*: mean gain over six benchmarks. *Right*: relative length change.

in training. We set temperature to 0.6, top-p to 0.95, and maximum generation length is 3,072 tokens. All reported numbers are three-run average results.

5.2 Main Results

Table 2 and Table 3 summarize the main RLVR and RLIF results. IDM consistently improves the corresponding GRPO and Intuitor baselines across both model scales and all six benchmarks. The largest gains occur on reasoning-intensive tasks: IDM-Seq improves MATH by 3.4/4.5 points under RLVR and 4.1/5.0 points under RLIF for the 3B/7B models, while also improving code reasoning on LCB and CRUX. Improvements on MMLU-Pro and AlpacaEval show that the gains do not come at the expense of broader language capabilities. Sequence-level modulation is generally slightly stronger and more stable than token-level modulation, suggesting that aggregating internal dynamics over a response provides a favorable trade-off between signal granularity and variance. Overall, the results show that cross-layer dynamics provide useful information beyond the scalar rewards used by both RLVR and RLIF.

5.3 Alternative Shaping Controls

We test whether IDM gains stem from generic reweighting by comparing DTS-based shaping with matched controls (length, entropy, shuffled DTS, random weights) under identical training settings. Results report mean improvement over six benchmarks. Figure 4 reveals a clear separation between DTS-based shaping and matched controls. Random weights and shuffled DTS consistently degrade performance, indicating that the gains are not due to arbitrary perturbations of the advantage. Length-based and entropy-based shaping perform even worse on average, despite increasing completion length, showing that longer outputs alone do not explain improvements. In contrast, DTS-based shaping yields substantial and consistent gains, with IDM-Tok improving by 1.99 points and IDM-Seq achieving the best overall performance at 2.29, while introducing only minimal length increase. Together, these results demonstrate that IDM's effectiveness is signal-specific rather than a generic effect of auxiliary reweighting, and that the benefit arises from capturing *where* and *how strongly* internal computation evolves, rather than from increased verbosity or noisy internal statistics.

5.4 Stability of DTS During Training

Because IDM uses DTS to modulate the learning signal, a natural concern is *depth hacking*: the policy might learn to artificially increase DTS without improving reasoning quality. We examine this possibility by tracking the percentile distribution of token-level DTS throughout training and comparing IDM with the corresponding RLIF baseline.

Figure 5 shows that the DTS distribution remains stable over training. Across percentiles, the coefficient of variation is below 8%, and the IDM curves closely track the RLIF baseline. This indicates that IDM does not systematically inflate the depth signal during optimization. Instead, DTS behaves as a stable read-only credit-assignment signal in our experiments: it affects how advantages are weighted, but we do not observe evidence that the policy learns to exploit the signal by increasing DTS itself.

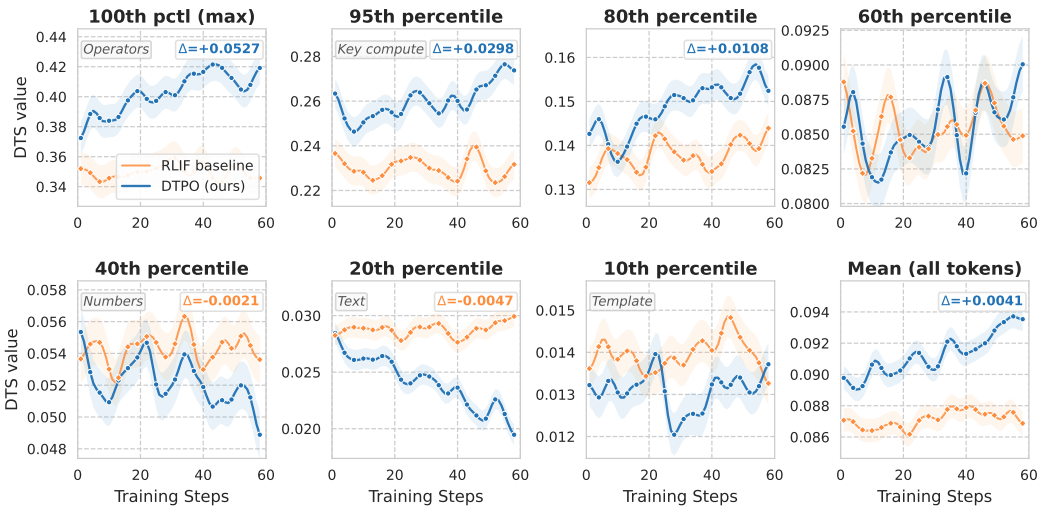

Figure 5: DTS percentile evolution during IDM training. The x-th percentile denotes the DTS value below which $x\%$ of generated tokens fall. Blue: IDM; orange: RLIF baseline; shaded area: $\pm\sigma$. The percentile curves remain stable and closely follow the baseline, providing no evidence of systematic depth inflation during training.

5.5 IDM Across Feedback Regimes

The main experiments above use verifiable rewards or self-certainty feedback. Because IDM operates after the learning signal is constructed, we next test whether the same modulation interface transfers to two different settings: self-generated test-time rewards and multi-turn environment feedback.

Table 4: IDM under additional feedback regimes.

Setting	Metric	Base	+ IDM
TTRL	pass@1	12.0	**13.0**
	majority@16	16.7	**17.8**
	best@16	31.3	**33.5**
Sokoban	mean (1–100)	16.7	**17.9**
	peak	46.1	**52.3**
	stop step	141	**304**

Test-time RL. We apply IDM to TTRL (Zuo et al., 2025), where majority-vote predictions on unlabeled test inputs provide pseudo-rewards for online policy optimization. We keep the TTRL reward construction and base objective unchanged and modulate the resulting learning signal. On Qwen2.5-Math-1.5B with AIME-TTT, IDM improves pass@1 from 12.0% to 13.0%, majority@16 from 16.7% to 17.8%, and best@16 from 31.3% to 33.5%.

Agentic RL. We further evaluate IDM on RAGEN Sokoban (Wang et al., 2025), where a multi-turn policy is trained from environment feedback. IDM improves mean success over the first 100 training steps from 16.7% to 17.9% and peak success from 46.1% to 52.3%. The observed stopping point also moves from step 141 to step 304, suggesting that IDM sustains effective optimization for substantially longer in this setting.

6 Conclusion

We introduce Internal Dynamics Modulation (IDM), a framework that incorporates model internal representation dynamics into RL by modulating existing policy advantages. IDM defines the Depth-Transition Score (DTS) from cross-layer hidden-state evolution and applies it at either sequence or token granularity. Across RLVR and RLIF, and six benchmarks spanning mathematical reasoning, code reasoning, language understanding, and instruction following, IDM consistently improves over RL baselines. Further analyses show that DTS is complementary to uncertainty and highlights reasoning-relevant internal dynamics. We additionally extend this modulation interface to test-time RL and multi-turn agentic RL, showing that IDM can operate across different sources of feedback. These results suggest that internal dynamics provide a useful signal for modulating RL in LLMs.

AI USE STATEMENT

Generative AI tools were used to assist with literature discovery, hypothesis refinement, experimental design discussion, code implementation and debugging, result analysis, and manuscript and figure preparation. All AI-assisted analyses, code, claims, citations, and reported experimental results were reviewed and verified by the authors. The authors take full responsibility for the final content of this work, including text, claims, code, and artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies reinforcement learning methods for large language models using publicly available models, benchmarks, and simulated environments. It does not involve human subjects, private or sensitive personal data, or deployment in high-stakes applications. We do not identify additional ethical risks beyond those generally associated with improving the capabilities of language models.

REPRODUCIBILITY STATEMENT

The IDM formulation and its sequence- and token-level implementations are specified in Section 4. Training and evaluation settings for the main RLVR and RLIF experiments are provided in Section 5.1. Additional internal-dynamics analyses, complete diagnostic results, online modulation details, and the TTRL and agentic RL configurations are reported in Appendices A and B. These sections document the models, datasets, sampling settings, modulation rules, and evaluation metrics used in the reported experiments.

REFERENCES

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Wei-Lin Chen, Liqian Peng, Tian Tan, Chao Zhao, Blake JianHang Chen, Ziqian Lin, Alec Go, and Yu Meng. Think deep, not just long: Measuring llm reasoning effort via deep-thinking tokens. *arXiv preprint arXiv:2602.13517*, 2026.

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

Róbert Csordás, Christopher D Manning, and Christopher Potts. Do language models use their depth efficiently? In *Proceedings of NeurIPS*, 2025.

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

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of EMNLP*, 2021.

Alex Gu, Baptiste Rozière, Hugh Leather, Armando Solar-Lezama, Gabriel Synnaeve, and Sida I. Wang. Cruxeval: A benchmark for code reasoning, understanding and execution. *arXiv preprint arXiv:2401.03065*, 2024.

Akshat Gupta, Jay Yeung, Gopala Anumanchipalli, and Anna Ivanova. How do llms use their depth? *arXiv preprint arXiv:2510.18871*, 2025.

Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. Lighteval: A lightweight framework for llm evaluation, 2023. URL https://github.com/huggingface/lighteval.

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

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. Process reward models that think. *arXiv preprint arXiv:2504.16828*, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Bill Yuchen Lin. ZeroEval: A Unified Framework for Evaluating Language Models, July 2024. URL https://github.com/WildEval/ZeroEval.

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991. doi: 10.1109/18.61115.

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

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. In *Proceedings of ICLR*, 2025.

OpenAI. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

OpenAI. Introducing gpt-4.1 in the api. OpenAI Official Blog, April 2025. URL https://openai.com/index/gpt-4-1/. Accessed: 2026-05-06.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Proceedings of NeurIPS*, 2022.

Qwen Team, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning. *arXiv preprint arXiv:2410.08146*, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. In *Proceedings of ACL*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of NeurIPS*, 2017.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of ACL*, 2024a.

Xucong Wang, Ziyu Ma, Yong Wang, Yuxiang Ji, Shidong Yang, Guanhua Chen, Pengkun Wang, and Xiangxiang Chu. Appo: Agentic procedural policy optimization. *arXiv preprint arXiv:2606.12384*, 2026.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of ICLR*, 2023.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *Proceedings of NeurIPS*, 2024b.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, et al. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

Jialiang Wu, Yi Shen, Sijia Liu, Yi Tang, Sen Song, Xiaoyi Wang, and Longjun Cai. Improve decoding factuality by token-wise cross layer entropy of large language models. In *Findings of NAACL*, 2025a.

Jianghao Wu, Yasmeen George, Jin Ye, Yicheng Wu, Daniel F. Schmidt, and Jianfei Cai. Spine: Token-selective test-time reinforcement learning with entropy-band regularization. *arXiv preprint arXiv:2511.17938*, 2025b.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Proceedings of NeurIPS*, 2023.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. In *Proceedings of NeurIPS*, 2025.

Yanzhi Zhang, Zhaoxi Zhang, Haoxiang Guan, Yilin Cheng, Yitong Duan, Chen Wang, Yue Wang, Shuxin Zheng, and Jiyan He. No free lunch: Rethinking internal feedback for llm reasoning. *arXiv preprint arXiv:2506.17219*, 2025.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. In *Proceedings of ICLR*, 2026.

Yuxin Zuo, Kaiyan Zhang, Shang Qu, Li Sheng, Xuekai Zhu, Biqing Qi, Youbang Sun, Ganqu Cui, Ning Ding, and Bowen Zhou. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*, 2025.

A Additional Diagnostic Results

A.1 Continuation-Sensitivity Coverage

For each reasoning prefix, we sample sibling continuations and measure the variability of their terminal outcomes. Table 5 reports the complete coverage results across selection budgets. Final-anchor DTS corresponds to the DTS construction used in the main method, while adjacent-layer DTS provides an alternative cross-layer comparison. Entropy and random ranking are included as output-side and signal-free references, respectively.

Table 5: **Continuation-sensitivity coverage across selection budgets.** Higher is better.

	K					
Selector	10	25	50	100	200	400
Qwen2.5-3B						
Final-anchor DTS	0.010	0.017	0.027	0.068	0.136	0.224
Adjacent-layer DTS	0.007	0.019	0.029	0.069	0.103	0.184
Entropy	0.022	0.029	0.059	0.106	0.148	0.241
Random	0.014	0.017	0.024	0.051	0.091	0.175
Qwen2.5-7B						
Final-anchor DTS	0.008	0.025	0.053	0.087	0.181	0.318
Adjacent-layer DTS	0.008	0.017	0.047	0.099	0.173	0.321
Entropy	0.020	0.027	0.040	0.064	0.116	0.253
Random	0.008	0.011	0.026	0.060	0.078	0.157

The two model scales exhibit different regimes. Entropy ranks continuation-sensitive prefixes more effectively for Qwen2.5-3B, whereas both DTS variants become more competitive for Qwen2.5-7B as the selection budget increases. In particular, final-anchor DTS reaches 0.181 coverage at $K = 200$, compared with 0.116 for entropy, while adjacent-layer DTS reaches the highest recorded coverage at $K = 400$.

A.2 Internal Dynamics Beyond Output Uncertainty

The analysis in Sec 3 shows that DTS is weakly correlated with output entropy at the token level. We further ask whether the two signals differ in their ability to identify intermediate reasoning states with sensitive downstream outcomes. For each selected prefix, we sample multiple sibling continuations and measure the variability of their terminal results, then compare prefixes ranked by DTS and by output entropy.

On Qwen2.5-7B, DTS captures more aggregate continuation sensitivity than entropy at moderate selection budgets. At $K = \{100, 200, 400\}$ selected prefixes, coverage is $\{0.087, 0.181, 0.318\}$ for DTS and $\{0.064, 0.116, 0.253\}$ for entropy. This advantage is most pronounced at $K = 200$, where DTS captures 56% more sensitivity mass. The smaller-model results are weaker, indicating that the complementarity between cross-layer dynamics and output uncertainty depends on the model regime.

We also compare soft and hard uses of the same signal during test-time aggregation. Soft DTS weighting retains all sampled responses and matches or improves full self-consistency, whereas selecting only the highest-DTS half substantially reduces MATH accuracy. On Qwen2.5-3B and 7B, soft weighting exceeds hard DTS selection by 5.0 and 8.5 points, respectively. This result supports using internal dynamics as a modulation signal rather than as a hard sample-selection criterion.

A.3 Soft and Hard DTS Aggregation

We additionally compare soft DTS weighting with hard response selection and alternative test-time aggregation rules. Each model–dataset setting contains 200 questions, with eight generated responses per question. Soft weighting retains the complete response pool and changes the relative contribution of each response, whereas hard-selection methods retain four responses.

Table 6: **Complete test-time aggregation results.** Accuracy (%). "Kept" denotes the number of responses retained by each aggregation rule.

Method	Kept	GSM8K 3B	GSM8K 7B	MATH 3B	MATH 7B
Self-consistency	8	89.0	93.0	66.5	75.5
Soft DTS	8	89.0	93.5	68.0	75.5
Hard DTS	4	86.0	92.0	63.0	67.0
Self-certainty	4	88.0	92.5	62.5	73.0
Log probability	4	87.0	92.5	63.0	71.5
Shortest	4	88.0	93.0	66.0	73.5
Longest	4	85.5	91.0	55.5	66.5
Random subset	4	87.5	94.0	64.0	72.0

Soft DTS weighting matches or improves full self-consistency while retaining all sampled responses. Hard DTS selection performs substantially worse on MATH, with gaps of 5.0 and 8.5 points relative to soft weighting for the 3B and 7B models, respectively. These results motivate using internal dynamics as a continuous modulation signal rather than converting DTS into a hard filtering rule.

B Online RL Details

B.1 Online Modulation Signal

The core representation-based experiments use DTS as the modulation signal. In the online TTRL and agentic settings, intermediate hidden states are not exposed by the rollout engine, so we use sampled-token surprisal as a lightweight observable signal. For a sampled token y_t, we define

$$s_t = -\log \pi_{\text{rollout}}(y_t \mid x, y_{<t}). \tag{9}$$

We normalize surprisal over valid response positions using

$$\tilde{s}_t = \frac{s_t - s_{\min}}{s_{\max} - s_{\min} + \epsilon}, \tag{10}$$

and map the normalized score to a positive modulation weight,

$$g_t = 0.5 + 0.5\tilde{s}_t. \tag{11}$$

The online weights are additionally normalized to unit mean over valid response tokens,

$$\hat{g}_t = \frac{g_t}{\frac{1}{T}\sum_{j=1}^{T} g_j}, \tag{12}$$

and the resulting advantage is

$$A'_t = A_t \hat{g}_t. \tag{13}$$

This within-response normalization is specific to the online experiments and is not part of the core DTS-based IDM formulation used in the main RLVR and RLIF experiments.

B.2 Test-Time RL Setup

For the TTRL experiment, we use Qwen2.5-Math-1.5B with the AIME-TTT configuration. Each test instance uses 64 sampled responses for pseudo-label voting and 32 sampled responses for policy optimization. Training runs for 80 epochs, corresponding to 240 optimizer steps. We report pass@1, majority@16, and best@16, and the values reported in Table 4 are averaged over the final five evaluation checkpoints.

B.3 Agentic RL Setup

For the agentic experiment, we use Qwen2.5-3B-Instruct in the RAGEN Sokoban environment with GRPO-based policy optimization and rollout-variance filtering. The planned optimization horizon is 400 steps. The baseline run triggers the training system's stopping condition at step 141, whereas the IDM run stops at step 304. Because the two trajectories have different observed lengths, we report the mean success rate over the common first 100 steps in addition to the maximum success rate observed over each complete trajectory.