Look Inward to Explore Outward: Learning Temperature Policy from LLM Internal States via Reinforcement Learning

Anonymous ACL submission

Abstract

Reinforcement Learning from Verifiable Rewards (RLVR) trains large language models (LLMs) from sampled trajectories, making decoding strategy a core component of learning rather than a purely inference-time choice. Sampling temperature directly controls the exploration–exploitation trade-off by modulating policy entropy, yet existing methods rely on static values or heuristic adaptations that are decoupled from task-level rewards. We propose *Introspective LLM*, a nested reinforcement learning framework that learns to control sampling temperature during generation. At each decoding step, the model introspects its own hidden state to select a temperature and samples the next token from the resulting distribution. Temperature and token policies are jointly optimized from downstream rewards using a coordinate ascent scheme. Experiments on mathematical reasoning benchmarks show that learned temperature policies outperform fixed and heuristic baselines, while exhibiting interpretable exploration behaviors aligned with reasoning uncertainty.

1 Introduction

Reinforcement Learning from Verifiable Rewards (RLVR) has significantly advanced LLM reasoning by optimizing models directly against task-level rewards, yielding substantial gains across domains such as mathematical reasoning (Shao et al., 2024; Guo et al., 2025a), code generation (Wang et al., 2024), and agentic tasks (Team et al., 2025; Zhang et al., 2025). Unlike supervised fine-tuning, where sampling primarily affects inference diversity, RLVR relies on the diversity of generated trajectories to drive policy improvement and sample efficiency. This dependency transforms the decoding strategy, particularly the sampling temperature τ, into a critical training-time component rather than a mere hyperparameter. By scaling the logits of the token distribution, temperature directly modulates policy entropy to govern the essential trade-off between exploring high-variance rollouts and exploiting stable generations.

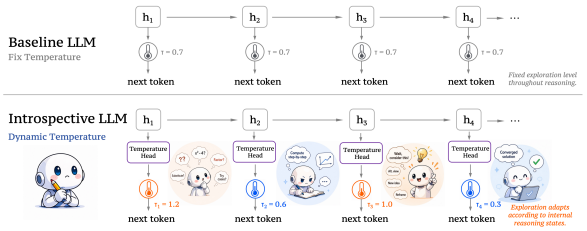

Figure 1: A baseline LLM uses a fixed τ at every decoding step; IntroLLM learns a per-token temperature policy conditioned on the current hidden state.

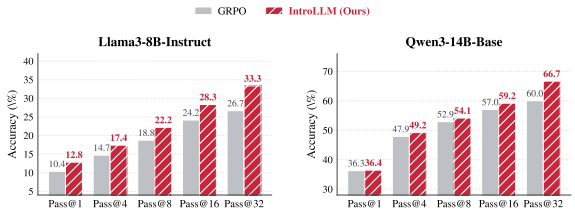

Figure 2: Pass@K accuracy of GRPO vs. IntroLLM on AIME 2024 at $K \in \{1, 4, 8, 16, 32\}$, for Llama3-8B-Instruct (left) and Qwen3-14B-Base (right). Gains over GRPO grow with K on both backbones and reach $+6.66$ points on Pass@32.

Despite the central role of temperature in RLVR, most existing approaches rely on either *static* temperature schedules or *heuristic* forms of adaptation. A single temperature is typically selected and applied uniformly across prompts, tokens, and training stages (Shao et al., 2024; Guo et al., 2025a; Yu et al., 2025). Recent empirical studies highlight the fragility of this design choice. For example, POLARIS (An et al., 2025) shows that the optimal sampling temperature varies substantially across base models and training phases, and that reinforcement learning progressively reduces model entropy, leading to a contraction of the effective exploration space. Similarly, Yang et al. (2025) demonstrates that exploration is not uniformly beneficial across

a sequence: high entropy is most valuable at early reasoning steps, while later tokens benefit from lower stochasticity.

More recent methods introduce dynamic adjustments based on hand-crafted signals, such as token entropy (Zhang et al., 2024; Zhuang et al., 2025), KL divergence (Chang et al., 2023), or predefined token categories (Zhu et al., 2024). While effective in specific regimes, these approaches hard-code assumptions about when and how exploration should occur, limiting their ability to adapt across tasks, prompts, and evolving model capabilities.

Crucially, heuristic temperature rules are *decoupled* from the learning objective: they do not receive direct feedback from task-level rewards and therefore cannot improve through experience. As a result, temperature selection remains a brittle design choice, even though it directly shapes the trajectories used for policy optimization. This raises a natural question: *Can a language model learn to control its own sampling behavior in order to maximize downstream reward?*

We answer this question by enabling the language model to "sense" its own generation context and modulate its exploration strategy accordingly. Specifically, we allow the model to observe its internal state during generation and decide how much stochasticity to inject at each step. This transforms sampling temperature from a rigid external hyperparameter into a context-aware policy that is optimized jointly with token generation under a unified reward signal.

We refer to this framework as *Introspective LLM (IntroLLM)*. Formally, it formulates adaptive temperature selection as a nested reinforcement learning problem, in which temperature decisions and token decisions are learned together under a shared reward signal. At each decoding step, generation proceeds under two coupled policies:

- a **temperature policy** $\pi_\phi(\tau_t \mid h_t, \tau_{t-1})$ that selects the sampling temperature based on the model's internal hidden state; and
- a **token policy** $\pi_\theta(y_t \mid h_t, \tau_t)$ that samples the next token from the rescaled distribution.

This nested structure enables the model to introspect its own uncertainty and generation context, dynamically deciding when to explore through higher-entropy sampling and when to exploit through more deterministic generation. Unlike prior approaches that rely on hand-crafted entropy thresholds or fixed schedules, these decisions are learned directly from data. Crucially, temperature choices are optimized with respect to the same task-level reward signal used to train the language model, allowing exploration strategies to adapt as the model's capabilities evolve during training and to vary across prompts and tokens.

Jointly learning temperature and token policies introduces nontrivial optimization challenges. To address this, we adopt a coordinate ascent training scheme where both policies are optimized via Group Relative Policy Optimization (GRPO) (Shao et al., 2024). We further model temperature selection as a *mixed discrete–continuous policy*: the policy first decides whether to change the temperature, and only then samples a new value if a change is made. This design allows the model to learn both *when* temperature adaptation is beneficial and *how much* adjustment is required, while avoiding unnecessary variance from frequent changes.

Experiments on mathematical reasoning benchmarks demonstrate that learned temperature policies consistently outperform both fixed-temperature baselines and heuristic adaptive methods. Beyond quantitative gains, analysis of the learned policies reveals interpretable patterns: higher temperatures are allocated to uncertain multi-step reasoning segments, while lower values are reserved for execution-heavy tokens such as numerical computation, factual retrieval, and final answer synthesis. Notably, these behaviors mirror insights from prior empirical studies (Wang et al., 2025b; An et al., 2025), but emerge naturally from reward-driven learning rather than manual design.

In summary, our main contributions are:

- We identify the decoding strategy, specifically sampling temperature, as a critical lever for exploration in RLVR and argue that it should be optimized rather than fixed.
- We formulate adaptive temperature selection as a nested reinforcement learning problem and propose a principled joint training framework.
- We introduce a mixed discrete–continuous temperature policy that learns when and how to adapt stochasticity, balancing exploration flexibility with training stability.
- We demonstrate consistent improvements over static and heuristic adaptive temperature methods on reasoning benchmarks, with interpretable learned behaviors.

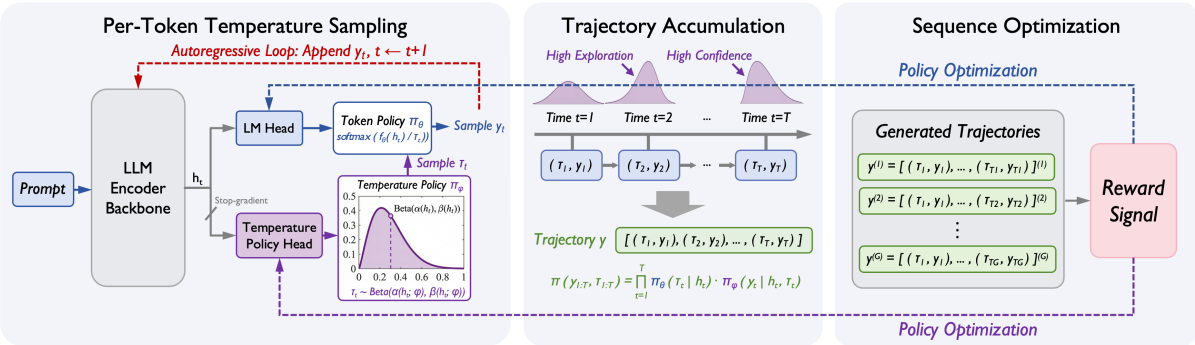

Figure 3: **Overview of the IntroLLM framework.** At each decoding step, a temperature policy π_ϕ observes the hidden state h_t and selects a sampling temperature τ_t, which then conditions the token policy π_θ to generate the next token y_t. Both policies are jointly optimized via reinforcement learning from verifiable task rewards.

2 Preliminary

Reinforcement Learning with Verifiable Rewards (RLVR). An autoregressive LLM defines a stochastic policy π_θ that samples tokens $y_t \sim \pi_\theta(y_t \mid h_t)$ to generate a sequence y given prompt x, aiming to maximize the expected reward $R(x, y)$ over a prompt distribution $\mathcal{D}$:

$$\max_\theta J(\theta) := \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)}[R(x, y)]. \quad (1)$$

To mitigate reward hacking in complex reasoning (Gao et al., 2023; Wang et al., 2025a), RLVR employs deterministic, rule-based signals for reliable feedback (Guo et al., 2025a). Optimization typically follows the policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(y_t \mid h_t)\, A_t\right], \quad (2)$$

where A_t is the advantage and T is the sequence length. To avoid the computational overhead of separate value models required by PPO (Schulman et al., 2017), Group Relative Policy Optimization (GRPO) (Shao et al., 2024) samples a group of G trajectories for each prompt and computes sequence-level advantages within the group:

$$A_i = \frac{R(x, y^{(i)}) - \text{mean}_{k\in 1:G} R(x, y^{(k)})}{\text{std}_{k\in 1:G} R(x, y^{(k)})}. \quad (3)$$

The policy is then updated via a clipped surrogate objective to ensure training stability:

$$\mathcal{L}(\theta) = \mathbb{E}\big[\min(r_t(\theta)A_t,\ \text{clip}_\epsilon(r_t(\theta))\, A_t)\big], \quad (4)$$

where $r_t(\theta) = \pi_\theta(y_t \mid h_t)/\pi_{\theta_{\text{old}}}(y_t \mid h_t)$ is the importance ratio and $\text{clip}_\epsilon(r) := \text{clip}(r, 1-\epsilon, 1+\epsilon)$.

Sampling Temperature. Temperature sampling (Ackley et al., 1985) is a widely used method to control the stochasticity of token selection. At step t, the model produces logits $\ell_t \in \mathbb{R}^{|V|}$ over the vocabulary V, which are rescaled by a temperature $\tau > 0$:

$$\pi_\theta(y_t = v \mid [x, y_{<t}]; \tau) = \frac{\exp(\ell_{t,v}/\tau)}{\sum_{v'\in V} \exp(\ell_{t,v'}/\tau)}, \quad (5)$$

where $v \in V$. Higher temperatures ($\tau > 1$) flatten the distribution, encouraging exploration by making less probable tokens more likely. Lower temperatures ($\tau < 1$) sharpen the distribution, favoring exploitation of high-probability tokens. In RLVR, temperature directly affects the diversity of trajectories, and therefore exploration.

3 Problem Formulation

Standard RLVR employs static or heuristic temperature schedules that cannot adapt to task heterogeneity. Empirical studies reveal three key limitations: (i) different reasoning steps require different exploration levels—reasoning pivots benefit from high temperature while factual tokens require low temperature (Cui et al., 2025b; Wang et al., 2025b); (ii) optimal temperature varies across prompts and difficulty levels; (iii) exploration requirements evolve during training as model capabilities improve (An et al., 2025).

We address these limitations by formulating temperature selection as a learnable sequential decision process. At each step t, a temperature policy $\pi_\phi(\tau_t \mid h_t, \tau_{t-1})$ selects sampling temperature τ_t based on hidden state h_t, while a token policy $\pi_\theta(y_t \mid h_t, \tau_t)$ generates the next token. The joint process factorizes as:

$$\pi(y, \tau \mid x) = \prod_{t=1}^{T} \big[\pi_\phi(\tau_t \mid h_t, \tau_{t-1}) \cdot \pi_\theta(y_t \mid h_t, \tau_t)\big], \quad (6)$$

where $y = (y_1, \ldots, y_T)$ and $\tau = (\tau_1, \ldots, \tau_T)$ are the token and temperature sequences. The objective is to maximize the expected reward:

$$J(\theta, \phi) = \mathbb{E}_{x \sim \mathcal{D}} \, \mathbb{E}_{\tau, y \sim \pi(\cdot|x)} \big[R(x, y) \big]. \quad (7)$$

4 Introspective LLM

As illustrated in Figure 3, IntroLLM instantiate the joint formulation through two coupled policies: (i) a **temperature policy** π_ϕ that observes hidden state h_t and selects sampling temperature τ_t, and (ii) a **token policy** π_θ that generates the next token y_t from a τ_t-modulated distribution. These operate sequentially at each decoding step. Training follows a standard RL pipeline: sample trajectories with both temperature and token decisions, evaluate via verifiable rewards, and update both policies jointly using policy gradients. The following subsections detail the token policy conditioning (§4.1), temperature policy design (§4.2), and joint optimization procedure (§4.3).

4.1 Token Policy with Temperature Conditioning

The token policy generates the next token from a distribution modulated by the temperature policy. Specifically, given hidden state h_t, the model produces pre-softmax logits $\ell_t \in \mathbb{R}^{|\mathcal{V}|}$, which are rescaled by the selected temperature τ_t to obtain the token distribution: $\pi_\theta(y_t \mid h_t, \tau_t) = \mathrm{Softmax}\left(\frac{\ell_t}{\tau_t}\right)$. This temperature-conditioned formulation enables adaptive entropy modulation, directly addressing the token-level heterogeneity identified in §3. During training, $\{\tau_t\}$ is treated as fixed parameters, allowing the token policy to focus on content generation while the temperature policy controls exploration intensity. The joint optimization procedure is detailed in §4.3.

4.2 Temperature Policy Design

The temperature policy $\pi_\phi(\tau_t \mid h_t)$ controls exploration by mapping the model's internal hidden states to dynamic sampling temperatures. This introspective mechanism enables the model to increase stochasticity at exploration steps while maintaining stability during exploitation. To balance expressive control with optimization stability, we formulate π_ϕ as a mixed discrete–continuous policy. This design allows the model to learn both *when* to adjust temperature and *how much* to adjust it, avoiding unnecessary variance from frequent changes while maintaining adaptation flexibility.

Architectural Parameterization. We implement the temperature policy as a lightweight MLP head f_ϕ branching from the final decoder layer. Given hidden representation $h_t \in \mathbb{R}^d$ at step t, the policy head projects it through a low-dimensional bottleneck to produce a 3-dimensional control vector $\mathbf{u}_t = [u_c, u_\alpha, u_\beta]^\top$:

$$\mathbf{u}_t = \mathbf{W}_2 \cdot \mathrm{ReLU}(\mathbf{W}_1 h_t + \mathbf{b}_1) + \mathbf{b}_2, \quad (8)$$

where $\mathbf{W}_1 \in \mathbb{R}^{\frac{d}{2} \times d}$ and $\mathbf{W}_2 \in \mathbb{R}^{3 \times \frac{d}{2}}$ are linear projections. This architecture ensures temperature decisions are conditioned on the full semantic and uncertainty context encoded in h_t, enabling the policy to balance exploration and exploitation.

Mixed Discrete–Continuous Action Space. The temperature selection proceeds through a two-stage stochastic process:

Stage 1: Update Decision. The policy first decides whether to adjust temperature via a Bernoulli random variable: $c_t \sim \mathrm{Bernoulli}(\sigma(u_c))$, where $\sigma(\cdot)$ is the sigmoid function. When $c_t = 0$, temperature remains constant ($\tau_t = \tau_{t-1}$), reducing variance by avoiding redundant adjustments when the current temperature is appropriate.

Stage 2: Value Sampling. Conditioned on $c_t = 1$, the policy samples a new temperature intensity z_t from a Beta distribution $\mathcal{B}(\alpha, \beta)$, which is naturally bounded in $[0, 1]$ and provides flexible bimodal and unimodal behaviors:

$$z_t \sim \mathrm{Beta}(\mathrm{softplus}(u_\alpha) + \epsilon, \mathrm{softplus}(u_\beta) + \epsilon), \quad (9)$$

where $\epsilon = 10^{-6}$ ensures numerical stability. The final sampling temperature is obtained via affine transformation to $[\tau_{\min}, \tau_{\max}]$: $\tau_t = \tau_{\min} + z_t \cdot (\tau_{\max} - \tau_{\min})$.

Joint Log-Likelihood. For policy gradient updates (§4.3), we compute the joint log-probability of a realized sample (c_t, z_t) as:

$$\log \pi_\phi(\tau_t \mid h_t) = \log P(c_t) + c_t \log p(z_t \mid \alpha, \beta), \quad (10)$$

where $c_t \in \{0, 1\}$ gates the continuous component. This factorization enables gradient computation while maintaining the coupling between discrete and continuous decisions.

4.3 Joint Policy Optimization

Updating θ and ϕ simultaneously couples their gradients and creates non-stationary dynamics, so we optimise by *coordinate ascent*. Each

iteration samples a group of G trajectories $\{(c^{(i)}, z^{(i)}, \tau^{(i)}, y^{(i)})\}_{i=1}^{G}$, scores them by the verifiable reward $R^{(i)}$, and computes the group-relative advantage A_i (Eq. (3)). The token policy is updated with τ held fixed and the temperature policy with y held fixed, using GRPO objectives that share A_i:

$$\mathcal{L}^{\text{token}}(\theta) = \mathbb{E}\big[\min(r_t^{\theta} A_i,\ \text{clip}_{\epsilon}(r_t^{\theta})\, A_i)\big], \quad (11)$$

$$\mathcal{L}^{\text{temp}}(\phi) = \mathbb{E}\big[\min(r_t^{\phi} A_i,\ \text{clip}_{\epsilon}(r_t^{\phi})\, A_i)\big], \quad (12)$$

where $\text{clip}_{\epsilon}(r) := \text{clip}(r, 1-\epsilon, 1+\epsilon)$ and the importance ratios are

$$r_t^{\theta} = \frac{\pi_{\theta}(y_t \mid h_t, \tau_t)}{\pi_{\theta_{\text{old}}}(y_t \mid h_t, \tau_t)}, \quad r_t^{\phi} = \frac{\pi_{\phi}(c_t, z_t \mid h_t)}{\pi_{\phi_{\text{old}}}(c_t, z_t \mid h_t)}. \quad (13)$$

At each iteration we recompute both log-probabilities of the cached actions (y_t, c_t, z_t) under the current parameters, then apply Eqs. (11)–(12). The shared advantage gives both policies consistent credit for the trajectory they jointly produced. Algorithm 1 lists the procedure (Appendix A); Appendix B proves convergence of this scheme under standard stochastic-gradient assumptions.

5 Experiments

Datasets and Benchmarks. We train all models on the MATH (Hendrycks et al., 2021) training set and evaluate on a comprehensive suite of reasoning tasks: AIME 2024, AMC 2023, MATH-500 (Lightman et al., 2023), Minerva Math (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), and Omni-Math (Gao et al., 2024).

Baselines. We compare IntroLLM against three categories of baselines: (i) Standard GRPO: GRPO training with fixed temperatures $\tau \in \{0.6, 1.0, 1.2\}$ to establish performance bounds for non-adaptive sampling; (ii) Heuristic Adaptive (EAD) (Yang et al., 2025): A pre-defined annealing schedule that monotonically decays temperature to balance early-stage exploration and late-stage convergence; (iii) Sequence-level Adaptive (TAMPO) (Dang et al., 2026): A learnable meta-policy that optimizes a static temperature for the entire sequence generation.

Implementation Details. All models are trained for 4 epochs with batch size 128, learning rate 1×10^{-6}, group size $G = 8$, and maximum response length 3072 tokens. For IntroLLM, we initialize the temperature policy head to a maximum-entropy state ($P(c_t = 1) = 0.5$, $z_t \sim \text{Uniform}(0, 1)$) with temperature bounds restricted to $[\tau_{\text{min}}, \tau_{\text{max}}] = [0.6, 1.5]$.

Evaluation Metrics. We report **Avg@8** (expected Pass@1 performance across multiple samples) and **Pass@8** to assess both reasoning accuracy and generation diversity. All methods use their respective temperature strategies with maximum response length 3072. Further details are provided in Appendix C.

5.1 Main Results

Mathematical Reasoning Performance. Table 1 compares IntroLLM with fixed-τ and adaptive baselines. Across the three fixed-τ GRPO variants no single value is optimal: the best choice swaps between $\tau \in \{0.6, 1.0, 1.2\}$ depending on benchmark and scale (e.g., τ=1.2 for AIME24-4B, τ=1.0 for AMC23-1.7B), and the best-worst gap reaches 1.6–2.0%.

Among adaptive methods, the heuristic annealing schedule EAD reaches 29.41%/39.66% average Avg@8 (1.7B/4B) and the sequence-level learned TAMPO reaches 28.28%/38.83%, while IntroLLM's token-level adaptive control attains 30.47%/40.91%, the best across both scales. Gains scale with both difficulty and model size. IntroLLM beats the strongest baseline by 4.17% on AIME24-4B and by ∼1–2% on Omni-Math-4B, but the margin shrinks to 0.7% on MATH-500 where headroom is smaller. Pass@8 gains over the second-best baseline likewise grow from +2.08% at 1.7B to +4.29% at 4B.

IntroLLM also broadens solution diversity, reaching Pass@8 of 61.79%/73.50% with a 5.0% jump on AMC23-4B (87.50%), and leads in average Avg@8 and Pass@8 at both scales, ranking first on most benchmark cells. A multi-seed study in Appendix F.1 confirms that the headline gains exceed seed-induced variance, with non-overlapping ±1 std intervals on most benchmarks.

Out-of-Domain Generalization. To assess whether the learned temperature policy captures domain-general reasoning patterns, we evaluate on three Out-of-Domain (OOD) tasks: GPQA-Diamond, MMLU-Pro, and HumanEval in Table 2. IntroLLM demonstrates consistent improvements across all OOD tasks. On the 1.7B model, IntroLLM surpasses the second-best baseline by 1.26% on GPQA, 0.23% on MMLU-Pro, and 1.82% on HumanEval. The 4B model shows similar patterns, with gains of 0.07% on GPQA,

Table 1: **Performance comparison on reasoning benchmarks.** For AIME24, AMC23, MATH-500, and Average, we report both **Avg@8** and **Pass@8**. **Bold** indicates best performance; Underline indicates the second best.

Method	AIME24		AMC23		MATH-500		Minerva	Olympiad	Omni	Average	
	Avg@8	Pass@8	Avg@8	Pass@8	Avg@8	Pass@8	Avg@8	Avg@8	Avg@8	Avg@8	Pass@8
Qwen3-1.7B-Base											
Base	1.67	10.00	18.12	65.00	31.30	74.80	7.12	12.69	8.23	13.19	49.93
GRPO ($\tau = 1.0$)	7.08	16.67	37.50	67.50	65.25	84.00	17.28	28.78	17.40	28.88	56.06
GRPO ($\tau = 0.6$)	6.67	20.00	35.00	67.50	64.60	85.60	15.17	28.23	16.66	27.72	57.70
GRPO ($\tau = 1.2$)	7.50	23.33	35.63	70.00	**68.35**	85.40	17.33	29.93	17.43	29.36	59.58
EAD (1.2→0.1)	7.08	23.33	37.50	70.00	67.03	85.80	17.33	29.64	17.90	29.41	59.71
TAMPO (0.6-1.5)	7.92	23.33	35.00	70.00	65.45	84.80	16.64	27.41	17.28	28.28	59.38
IntroLLM (Ours)	**9.58**	**26.67**	**39.06**	**72.50**	67.53	**86.20**	**18.29**	**29.95**	**18.43**	**30.47**	**61.79**
Qwen3-4B-Base											
Base	6.67	20.00	27.19	72.50	48.68	86.40	12.50	24.30	14.94	22.38	59.63
GRPO ($\tau = 1.0$)	14.17	33.33	50.00	80.00	79.10	92.00	21.51	42.36	23.88	38.50	68.44
GRPO ($\tau = 0.6$)	11.67	30.00	50.94	82.50	77.60	91.40	21.09	41.06	23.41	37.63	67.97
GRPO ($\tau = 1.2$)	14.58	30.00	51.88	82.50	80.03	92.00	22.61	44.12	24.35	39.60	68.17
EAD (1.2→0.1)	14.17	36.67	51.88	77.50	80.00	92.20	**23.90**	43.69	24.32	39.66	68.79
TAMPO (0.6-1.5)	15.00	33.33	50.62	82.50	78.23	91.80	22.79	42.43	23.92	38.83	69.21
IntroLLM (Ours)	**19.17**	**40.00**	**52.50**	**87.50**	**80.73**	**93.00**	23.48	**44.25**	**25.34**	**40.91**	**73.50**

Table 2: **Out-of-domain generalization performance.** We report accuracy (%) for GPQA-Diamond (GPQA-D) and MMLU-Pro, and Pass@1 (%) for HumanEval.

Model / Method	GPQA-D	MMLU-Pro	HumanEval
Qwen3-1.7B-Base			
Base	21.72	15.49	28.05
GRPO ($\tau = 1.0$)	28.48	28.39	33.54
EAD (1.2→0.1)	29.36	28.66	33.54
Tampo (0.6-1.5)	27.08	27.16	35.98
IntroLLM (Ours)	**30.62**	**28.89**	**37.80**
Qwen3-4B-Base			
Base	26.77	25.06	66.46
GRPO ($\tau = 1.0$)	35.20	36.60	75.61
EAD (1.2→0.1)	38.19	36.22	76.22
Tampo (0.6-1.5)	36.05	37.23	76.22
IntroLLM (Ours)	**38.26**	**37.72**	**76.83**

0.49% on MMLU-Pro, and 0.61% on HumanEval. Consistent gains across domains, from scientific reasoning to code generation, demonstrate that IntroLLM's token-level temperature adaptation captures generalizable exploration strategies.

Computational Efficiency. Table 3 demonstrates that IntroLLM introduces negligible computational overhead. The temperature policy head adds only 2.1M parameters (0.122% increase) to Qwen3-1.7B-Base, resulting in minimal memory footprint during training and inference. We measure FLOPs and generation throughput across context lengths from 1k to 24k tokens. FLOPs remain virtually identical to the baseline across all contexts. Throughput degradation is minimal: approximately 2–3 tokens/s at the common 4k context length, with

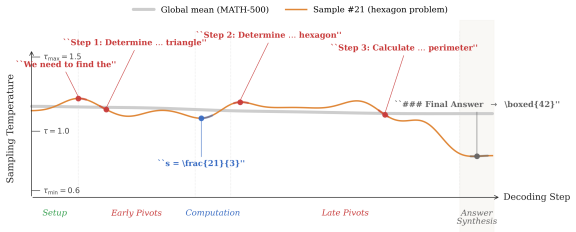

Figure 4: **Learned temperature patterns.** Gray dashed line: global average across MATH-500 showing natural annealing. Orange line: individual problem showing "reasoning rhythm" with peaks at logical pivots and valleys during computation.

negligible impact even at 24k tokens. Training time also remains efficient: at the 4B scale, IntroLLM takes 3h 41min versus 3h 14min for GRPO, an extra 26 minutes. Appendix F.4 reports the full 1.7B/4B/14B comparison and shows that the relative overhead does not worsen as the base model is scaled up. These results demonstrate that IntroLLM achieves substantial reasoning improvements with minimal computational cost, making it practical for large-scale deployment.

5.2 Further Analysis

Temperature Dynamics During Reasoning. To understand IntroLLM's inference-time behavior, we visualize per-token temperatures on MATH-500 (Figure 4). Globally (averaged across all problems), the model exhibits a natural annealing pattern, shifting from early exploration to late exploitation. At the individual problem level, IntroLLM develops a "reasoning rhythm"—modulating temperature

Table 3: **Efficiency profile on Qwen3-1.7B-Base.** We compare the computational cost (FLOPs) and generation throughput (tokens/s) across context lengths from 1k to 24k. The **Param. Δ** column indicates the additional parameter overhead introduced by IntroLLM.

Metrics	Method	Param. Δ	1k	2k	4k	8k	16k	24k
FLOPs	Default Sampling	–	4.85e+15	8.08e+15	1.45e+16	2.75e+16	5.33e+16	7.92e+16
	IntroLLM (Ours)	+0.122%	4.85e+15	8.09e+15	1.46e+16	2.75e+16	5.34e+16	7.93e+16
Throughput	Default Sampling	–	146.37	145.39	144.14	143.44	142.14	139.87
	IntroLLM (Ours)	+0.122%	144.08	142.58	141.71	140.22	136.01	133.37

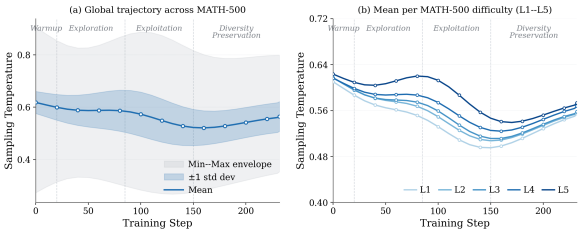

Figure 5: **Temperature evolution during training.** (Left) Global mean and extrema trajectories across MATH-500. (Right) Mean values broken down by MATH-500 difficulty level. The policy follows an emergent non-monotonic cycle of exploration, exploitation, and diversity preservation, distinct from hand-crafted annealing schedules.

through multiple peaks and troughs, with higher temperatures at logical pivots and lower temperatures during computation and answer synthesis.

Temperature Evolution During Training. To examine how IntroLLM modulates exploration over training, we visualise the global temperature trajectory in Figure 5. The policy follows a non-monotonic adaptation pattern that contrasts sharply with hand-crafted annealing schedules: it first raises temperature to enable broad exploration, then cools to consolidate high-reward patterns, and finally rises again in late training to preserve solution diversity. This emergent "exploration–exploitation–diversity" cycle is absent from predefined schedules and helps explain the superior Pass@8 we observe. Two complementary slices of the learned policy—its dependence on problem difficulty and on token-level semantics—are deferred to Appendix F.2.

Mitigating Entropy Collapse. We monitor policy entropy and rewards across IntroLLM and three baselines to assess training stability (Figure 6). Standard GRPO suffers from rapid entropy collapse, leading to premature convergence. In contrast, IntroLLM maintains significantly higher entropy during early training, facilitating more thorough exploration of the reasoning space. This sustained exploration translates to consistently higher average rewards and superior validation performance (Mean@4 and Best@4) than GRPO, TAMPO, and EAD. These results demonstrate that adaptive temperature control effectively mitigates entropy collapse and stabilizes RLVR training.

5.3 Ablation Study

We ablate two design choices of the temperature policy, control granularity (prompt-level vs. token-level) and update frequency (selective vs. every-token), by comparing three variants on AIME 2024 and MATH-500 (Figure 7; full per-benchmark numbers in Appendix F.3). A *prompt-level* variant assigns a single temperature to the entire response based only on the prompt. This coarse-grained control cannot adapt within a sequence and lags our method by 2.81%/1.38% in average Avg@8 on the two scales. Switching to an *always-update* variant that resamples temperature at every token without the Bernoulli trigger removes the within-sequence rigidity but injects excessive variance into the policy gradient, so much so that on 4B it actually underperforms the prompt-level baseline by 1.55% in average Avg@8. Our discrete-continuous mixed action space combines the best of both. By learning *when* to change temperature (Bernoulli) and *how much* (Beta), it reaches 38.72%/50.80% average Avg@8 (1.7B/4B), beating prompt-level by 2.81%/1.38% and always-update by 0.88%/2.93%. Together these results show that token-level granularity and selective updates are both load-bearing. Granularity lets the policy track within-sequence reasoning dynamics, and selectivity keeps that flexibility from destabilising learning.

6 Related Work

RLVR and Exploration Control. LLM post-training has evolved from outcome-level preference modeling (Christiano et al., 2017; Ouyang et al., 2022; Rafailov et al., 2023) to process-level reasoning rewards (Lightman et al., 2023; Cui et al.,

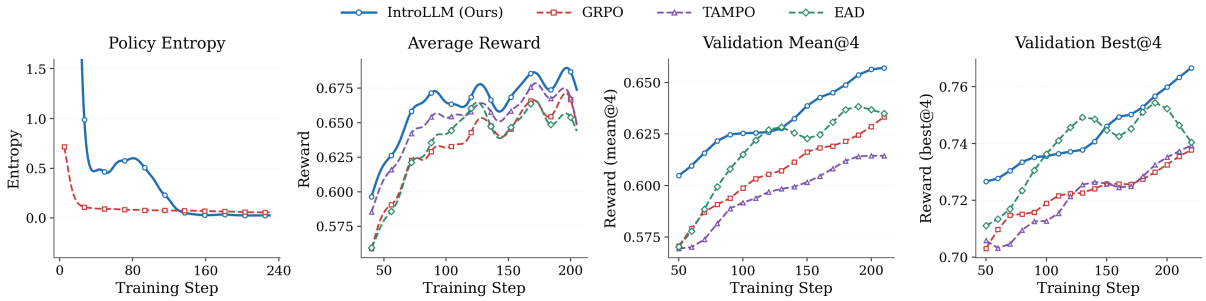

Figure 6: **Training curves comparing IntroLLM with GRPO, TAMPO, and EAD.** (Left to right) Policy Entropy on Qwen3-4B-Base; Average Reward, Validation Mean@4, and Validation Best@4 on Qwen3-1.7B-Base. IntroLLM maintains higher policy entropy and sustained exploration than GRPO, leading to higher average rewards and superior validation performance against all three baselines.

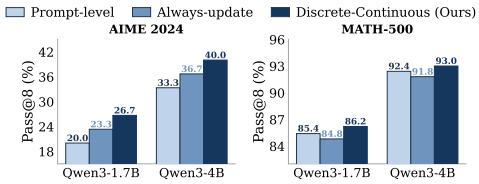

Figure 7: **Ablation of temperature-policy design choices on AIME 2024 and MATH-500.** Three variants are compared at both Qwen3 scales (1.7B and 4B): *prompt-level* (one τ per response), *always-update* (resample τ at every token), and our *discrete–continuous* mixed action space (Bernoulli trigger + Beta sampling). Full per-benchmark numbers including AMC23 are in Table 7 (Appendix F.3).

2025a; Choudhury, 2025). To mitigate reward hacking in complex tasks (Gao et al., 2023; Wang et al., 2025a), Reinforcement Learning from Verifiable Rewards (RLVR) (Jaech et al., 2024; Guo et al., 2025a) has emerged as the standard for reasoning-oriented models, driving research into scalable optimization and reward design (Yu et al., 2025; Guo et al., 2025b; Zheng et al., 2025; Chen et al., 2025a; Yuan et al., 2024; Zhao et al., 2025). However, the sparse feedback in RLVR makes discovering valid reasoning paths particularly challenging. While traditional exploration relies on entropy maximization (Ziebart et al., 2008; Haarnoja et al., 2018) or intrinsic motivation (Burda et al., 2018; Pathak et al., 2017), recent RLVR studies address this by localizing exploration to decision-critical tokens (Cheng et al., 2025; Cui et al., 2025b; Wang et al., 2025c; He et al., 2025; Wang et al., 2025d), optimizing Pass@k objectives (Chen et al., 2025b; Walder and Karkhanis, 2025), or incorporating semantic diversity signals (Hu et al., 2025; Li et al., 2025). Unlike these heuristic-based approaches, IntroLLM treats exploration control itself as a learnable, reward-driven policy.

Adaptive Temperature Sampling. While temperature effectively modulates policy entropy, static schedules fail to account for token-level heterogeneity, training-stage drift, and prompt-specific dynamics. Empirical studies confirm that optimal exploration requirements vary significantly across training stages and token positions (An et al., 2025; Yang et al., 2025). Recent dynamic strategies either incorporate parametric control via supervised fine-tuning (Wang et al., 2025e) or adjust distribution through heuristics such as entropy or KL-divergence (Zhang et al., 2024; Zhuang et al., 2025; Chang et al., 2023). Unlike these heuristics, IntroLLM optimizes temperature as a learnable policy from task-level rewards, enabling fully automated, reward-driven exploration.

7 Conclusion

In this work, we studied the role of sampling temperature in reinforcement learning with verifiable rewards and argued that exploration control should be treated as a first-class learning problem rather than a fixed heuristic. We introduced a joint formulation in which sampling temperature is modeled as a learnable policy and optimized jointly with token generation using the same task reward. This framework enables context-aware, reward-driven exploration that adapts across prompts, token positions, and training stages. Empirical results demonstrate that learning temperature leads to more effective exploration, improved reasoning performance, and more stable training compared to fixed or heuristically scheduled temperatures. Beyond the specific instantiation studied here, our formulation suggests a broader perspective on decoding as a controllable and learnable component of RL-based language model training, opening avenues for principled optimization of other generation-time decisions.

Limitations

Our study has several limitations. First, we evaluate IntroLLM exclusively on mathematical and code-style reasoning benchmarks where rewards are verifiable; the framework's effectiveness in open-ended generation tasks with noisier reward signals remains untested. Second, the temperature policy is conditioned on the last-layer hidden state; alternative conditioning signals (e.g., intermediate-layer representations or attention statistics) are not explored. Third, the temperature bounds $[\tau_{\min}, \tau_{\max}] = [0.6, 1.5]$ and the maximum-entropy initialization were selected based on preliminary exploration; we have not systematically studied their sensitivity. Finally, our analysis focuses on aggregate metrics (Avg@8, Pass@8); a more granular examination of how learned temperature interacts with specific reasoning failure modes is left to future work.

Ethics Statement

This work contributes to the study of reinforcement learning–based post-training for large language models by proposing a principled framework for adaptive exploration control. By learning when and how much to explore during generation, our method can improve reasoning reliability and training efficiency in settings with verifiable rewards. This has the potential to benefit applications that rely on structured reasoning and correctness, such as mathematics, scientific problem solving, code generation, and formal verification, where discovering valid solution trajectories under sparse feedback is critical.

From a methodological perspective, treating decoding parameters as learnable control variables may encourage more transparent and systematic design of RL-based language model training, reducing reliance on brittle heuristics and extensive manual tuning. Improved exploration efficiency could also lower the computational cost of RLVR training, potentially reducing energy consumption and resource barriers for research and deployment.

At the same time, our work does not directly address broader risks associated with large language models, such as misuse, bias, or hallucination outside verifiable settings. While adaptive exploration can improve reasoning diversity, it may also increase the generation of incorrect or unsafe intermediate outputs if applied without reliable verification signals. As a result, the proposed framework is most appropriate in domains where correctness can be automatically checked, and caution is required when extending it to open-ended or safety-critical applications.

References

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. 1985. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and 1 others. 2025. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. *URL https://hkunlp. github. io/blog/2025/Polaris*.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. 2018. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*.

Chung-Ching Chang, David Reitter, Renat Aksitov, and Yun-Hsuan Sung. 2023. Kl-divergence guided temperature sampling. *arXiv preprint arXiv:2306.01286*.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, and 1 others. 2025a. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*.

Zhipeng Chen, Xiaobo Qin, Youbin Wu, Yue Ling, Qinghao Ye, Wayne Xin Zhao, and Guang Shi. 2025b. Pass@ k training for adaptively balancing exploration and exploitation of large reasoning models. *arXiv preprint arXiv:2508.10751*.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. 2025. Reasoning with exploration: An entropy perspective. *arXiv preprint arXiv:2506.14758*.

Sanjiban Choudhury. 2025. Process reward models for llm agents: Practical framework and directions. *arXiv preprint arXiv:2502.10325*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, and 1 others. 2025a. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, and 1 others. 2025b. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

Haoran Dang, Cuiling Lan, Hai Wan, Xibin Zhao, and Yan Lu. 2026. Temperature as a meta-policy: Adaptive temperature in LLM reinforcement learning. In *The Fourteenth International Conference on Learning Representations*.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, and 1 others. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, and 1 others. 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.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, and 1 others. 2025. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*.

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*.

Zican Hu, Shilin Zhang, Yafu Li, Jianhao Yan, Xuyang Hu, Leyang Cui, Xiaoye Qu, Chunlin Chen, Yu Cheng, and Zhi Wang. 2025. Diversity-incentivized exploration for versatile reasoning. *arXiv preprint arXiv:2509.26209*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857.

Tianjian Li, Yiming Zhang, Ping Yu, Swarnadeep Saha, Daniel Khashabi, Jason Weston, Jack Lanchantin, and Tianlu Wang. 2025. Jointly reinforcing diversity and quality in language model generations. *arXiv preprint arXiv:2509.02534*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

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, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Christian Walder and Deep Karkhanis. 2025. Pass@ k policy optimization: Solving harder reinforcement learning problems. *arXiv preprint arXiv:2505.15201*.

Chaoqi Wang, Zhuokai Zhao, Yibo Jiang, Zhaorun Chen, Chen Zhu, Yuxin Chen, Jiayi Liu, Lizhu Zhang, Xiangjun Fan, Hao Ma, and 1 others. 2025a. Beyond reward hacking: Causal rewards for large language model alignment. *arXiv preprint arXiv:2501.09620*.

Jiakang Wang, Runze Liu, Fuzheng Zhang, Xiu Li, and Guorui Zhou. 2025b. Stabilizing knowledge, promoting reasoning: Dual-token constraints for rlvr. *arXiv preprint arXiv:2507.15778*.

Junqiao Wang, Zeng Zhang, Yangfan He, Zihao Zhang, Xinyuan Song, Yuyang Song, Tianyu Shi, Yuchen Li, Hengyuan Xu, Kunyu Wu, and 1 others. 2024. Enhancing code llms with reinforcement learning in code generation: A survey. *arXiv preprint arXiv:2412.20367*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, and 1 others. 2025c. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, and 1 others. 2025d. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*.

Zhichao Wang, Dongyang Ma, Xinting Huang, Deng Cai, Tian Lan, Jiahao Xu, Haitao Mi, Xiaoying Tang, and Yan Wang. 2025e. The end of manual decoding: Towards truly end-to-end language models. *arXiv preprint arXiv:2510.26697*.

Chenghao Yang, Lin Gui, Chenxiao Yang, Victor Veitch, Lizhu Zhang, and Zhuokai Zhao. 2025. Let it calm: Exploratory annealed decoding for verifiable reinforcement learning. *arXiv preprint arXiv:2510.05251*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. 2024. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, and 1 others. 2025. The landscape of agentic reinforcement learning for llms: A survey. *arXiv preprint arXiv:2509.02547*.

Shimao Zhang, Yu Bao, and Shujian Huang. 2024. Edt: Improving large language models' generation by entropy-based dynamic temperature sampling. *arXiv preprint arXiv:2403.14541*.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. 2025. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, and 1 others. 2025. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*.

Yuqi Zhu, Jia Li, Ge Li, YunFei Zhao, Zhi Jin, and Hong Mei. 2024. Hot or cold? adaptive temperature sampling for code generation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 437–445.

Haomin Zhuang, Yujun Zhou, Taicheng Guo, Yue Huang, Fangxu Liu, Kai Song, and Xiangliang Zhang. 2025. Exploring multi-temperature strategies for token-and rollout-level control in rlvr. *arXiv preprint arXiv:2510.08892*.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, and 1 others. 2008. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA.

A Training Algorithm

We expand on the rollout, recomputation, and credit-assignment steps below; this complements Algorithm 1 and the compact equations in §4.3.

Joint Rollout. During rollout, generation proceeds via a nested stochastic process. At each step t, the temperature policy samples (c_t, z_t) to determine τ_t, and the token policy samples y_t conditioned on τ_t. We persist the trajectory $\mathcal{T} = \{(c_t, z_t, \tau_t, y_t)\}_{t=1}^{T}$ in the experience buffer, where $\{y_t\}$ enables reward evaluation and $\{(c_t, z_t, \tau_t)\}$ enables likelihood recomputation.

On-Policy Likelihood Recomputation. To compute policy gradients, we must evaluate the log-probabilities of the actions taken during rollout under the *current* policy parameters. During each training iteration we perform a single forward pass to recompute likelihoods for both policies. (i) Token policy: $\log \pi_\theta(y_t \mid h_t, \tau_t)$ is obtained by scaling token logits with the cached temperatures $\{\tau_t\}$ from the stored trajectory, so gradients reflect the exact exploration context of the original rollout. (ii) Temperature policy: $\log \pi_\phi(c_t, z_t \mid h_t)$ is computed for the recorded temperature actions, reinforcing the exploration strategy based on its sequence-level rewards. Both likelihoods are then used in the GRPO objective to update the respective policies via coordinate ascent.

Algorithm 1 IntroLLM training (one GRPO iteration)

Require: Token policy π_θ, temperature policy π_ϕ with head f_ϕ, prompt batch, group size G, bounds $[\tau_{\min}, \tau_{\max}]$, clip ϵ.

for each prompt x **do**
  // Joint rollout: sample G trajectories under (π_ϕ, π_θ)
  for $i = 1, \ldots, G$ **do**
    $\tau_{i,0} \leftarrow 1.0$
    for $t = 1, \ldots, T$ **do**
      $[u_c, u_\alpha, u_\beta] \leftarrow f_\phi(h_{i,t})$; $c_{i,t} \sim \mathrm{Bernoulli}(\sigma(u_c))$
      if $c_{i,t} = 1$ **then**
        $z_{i,t} \sim \mathrm{Beta}(\mathrm{sp}(u_\alpha), \mathrm{sp}(u_\beta))$; $\tau_{i,t} \leftarrow \tau_{\min} + z_{i,t}(\tau_{\max} - \tau_{\min})$
      else
        $\tau_{i,t} \leftarrow \tau_{i,t-1}$
      end if
      $y_{i,t} \sim \pi_\theta(\cdot \mid h_{i,t}, \tau_{i,t})$
    end for
    $R_i \leftarrow \mathrm{VerifiableReward}(x, y^{(i)})$
  end for
  Compute group-relative advantage A_i (Eq. 3) from $\{R_k\}_{k=1}^G$
  // Coordinate ascent: update each policy with the other held fixed
  Update θ via $\mathcal{L}^{\text{token}}$ (Eq. 11) on $\{y_t\}$
  Update ϕ via $\mathcal{L}^{\text{temp}}$ (Eq. 12) on $\{(c_t, z_t)\}$
end for
$\theta_{\text{old}} \leftarrow \theta, \quad \phi_{\text{old}} \leftarrow \phi$

Coordinate Ascent Optimization. We optimize θ and ϕ via coordinate ascent, alternating updates within each GRPO iteration. Given a group of trajectories $\mathcal{T} = \{(c^{(i)}, z^{(i)}, \tau^{(i)}, y^{(i)})\}_{i=1}^G$:

1. *Token policy update.* Fixing τ as constants, we update θ to maximise reward conditioned on the realised exploration:

$$\theta \leftarrow \arg\max_\theta \mathbb{E}_{\mathcal{T}}\Big[\tfrac{1}{G}\sum\nolimits_{i,t} \mathcal{L}_{i,t}^{\text{token}}(\theta \mid \tau_t^{(i)})\Big],$$

where $\mathcal{L}^{\text{token}}$ is the GRPO clipped objective from Equation (4) applied to token actions y_t, using the importance ratio $r_t^\theta = \pi_\theta(y_t|h_t, \tau_t)/\pi_{\theta_{\text{old}}}(y_t|h_t, \tau_t)$.

2. *Temperature policy update.* Fixing tokens y as constants, we update ϕ to optimise the exploration strategy that produced those rewards:

$$\phi \leftarrow \arg\max_\phi \mathbb{E}_{\mathcal{T}}\Big[\tfrac{1}{G}\sum\nolimits_{i,t} \mathcal{L}_{i,t}^{\text{temp}}(\phi \mid y_t^{(i)})\Big],$$

where $\mathcal{L}^{\text{temp}}$ applies the same GRPO objective (Equation (4)) to temperature actions (c_t, z_t) with importance ratio $r_t^\phi = \pi_\phi(c_t, z_t|h_t)/\pi_{\phi_{\text{old}}}(c_t, z_t|h_t)$.

Both updates share the trajectory-level advantage A_i (Equation (3)), computed from the final reward $R(x, y^{(i)})$ and normalised across the group. This shared advantage ensures consistent credit assignment: if a trajectory achieves high reward, both the temperature decisions and the token choices that produced it are reinforced proportionally. By isolating gradient flows in this way, the alternating scheme mitigates the non-stationarity of simultaneous updates and yields stable convergence.

B Convergence of the Coordinate-Ascent Training Scheme

We provide a formal convergence analysis of the coordinate-ascent update used by IntroLLM (Algorithm 1). Let $z = (\theta, \phi) \in \mathbb{R}^{d_\theta + d_\phi}$, and let $J(\theta, \phi)$ denote the expected verifiable reward under the joint policy $\pi(y, \tau \mid x) = \prod_t \pi_\phi(\tau_t \mid h_t)\, \pi_\theta(y_t \mid h_t, \tau_t)$. The stochastic gradient at iteration k is $\hat{g}_k = (\hat{g}_{\theta,k}, \hat{g}_{\phi,k})$, the *unclipped* GRPO score-function policy-gradient estimator from a common group of G rollouts (the clipped surrogate used in Algorithm 1 is treated as a perturbation of $\hat{g}_k$ in the PPO-clipping paragraph below). The update is $z_{k+1} = z_k + \eta_k\, \hat{g}_k$. Let $\mathcal{F}_k := \sigma(z_0, \hat{g}_0, \ldots, \hat{g}_{k-1})$ be the natural filtration, with z_k $\mathcal{F}_k$-measurable and η_k deterministic.

Assumptions.

(A1) *Block L-smoothness:* $\|\nabla J(z') - \nabla J(z)\| \le L\,\|z' - z\|$ for all z, z'.

(A2) *Conditional unbiasedness with bounded variance:* $\mathbb{E}[\hat{g}_k \mid \mathcal{F}_k] = \nabla J(z_k)$ and $\mathbb{E}[\|\hat{g}_k - \nabla J(z_k)\|^2 \mid \mathcal{F}_k] \le \sigma^2 < \infty$.

(A3) *Bounded objective, deterministic initialisation:* $J^\star := \sup_z J(z) < \infty$ (automatic since the verifiable reward is binary, so $J \in [0, 1]$); z_0 is deterministic. If z_0 is random, replace $J(z_0)$ by $\mathbb{E}[J(z_0)]$ in all bounds below.

(A4) *Robbins–Monro step sizes:* $\eta_k > 0$ and $\eta_k \le 1/L$ for every k; $\sum_k \eta_k = \infty$; $\sum_k \eta_k^2 < \infty$. Positivity and the cap $\eta_k \le 1/L$ are needed for the descent inequality, the monotone-series argument, and the minimum-bound step. (Used in Part 1 only.)

Theorem B.1 (Convergence of IntroLLM coordinate ascent)**.** *The iterates $\{z_k\}$ of the IntroLLM update rule satisfy:*

1. *(*Subsequence stationarity in expectation.*) Under (A1)–(A4), $\liminf_{k\to\infty} \mathbb{E}[\|\nabla J(z_k)\|^2] = 0$: there exists a subsequence $\{k_j\}$ along which $\mathbb{E}[\|\nabla J(z_{k_j})\|^2] \to 0$. This is* not *iterate convergence,* not *almost-sure stationarity, and* not *convergence of the full sequence $\mathbb{E}[\|\nabla J(z_k)\|^2]$.*
2. *(*Best-iterate finite-time rate.*) Under (A1)–(A3) and the schedule $\eta_k = \eta_0/\sqrt{k+1}$ with $\eta_0 \in (0, 1/L]$ (which does* not *satisfy (A4) on its own, since $\sum 1/(k+1) = \infty$), for every $K \geq 2$,*

$$\begin{aligned} &\min_{0\leq k\leq K-1} \mathbb{E}[\|\nabla J(z_k)\|^2] \\ &\leq \frac{2\big(J^\star - J(z_0)\big) + L\sigma^2\eta_0^2\big(1+\log K\big)}{2\,\eta_0\,\big(\sqrt{K}-1\big)} \\ &= O\big((\log K)/\sqrt{K}\big). \end{aligned} \tag{14}$$

The bound is on the minimum *expected squared gradient norm over the first K iterates; the locator of the minimiser is not specified. The hidden O-constant depends on $J^\star - J(z_0)$, L, σ^2, and η_0; in the per-block extension it additionally depends on $1/c_{\min}$ and $c_{\max}^2/c_{\min}$.*

Proof. Step 1 (descent lemma). By (A1), J is L-smooth, so $J(z') \geq J(z) + \langle \nabla J(z), z' - z\rangle - (L/2)\|z' - z\|^2$. Setting $z = z_k$, $z' = z_{k+1} = z_k + \eta_k \hat{g}_k$,

$$J(z_{k+1}) \;\geq\; J(z_k) + \eta_k \langle \nabla J(z_k), \hat{g}_k\rangle - \tfrac{L\eta_k^2}{2}\,\|\hat{g}_k\|^2. \tag{15}$$

Write $\hat{g}_k = \nabla J(z_k) + \xi_k$ with $\xi_k := \hat{g}_k - \nabla J(z_k)$. By (A2), $\mathbb{E}[\xi_k \mid \mathcal{F}_k] = 0$ and $\mathbb{E}[\|\xi_k\|^2 \mid \mathcal{F}_k] \leq \sigma^2$. Since $\nabla J(z_k)$ is $\mathcal{F}_k$-measurable, $\mathbb{E}[\langle \nabla J(z_k), \xi_k\rangle \mid \mathcal{F}_k] = \langle \nabla J(z_k), \mathbb{E}[\xi_k \mid \mathcal{F}_k]\rangle = 0$, so

$$\begin{aligned} \mathbb{E}[\|\hat{g}_k\|^2 \mid \mathcal{F}_k] &= \|\nabla J(z_k)\|^2 + \mathbb{E}[\|\xi_k\|^2 \mid \mathcal{F}_k] \\ &\leq \|\nabla J(z_k)\|^2 + \sigma^2. \end{aligned} \tag{16}$$

Taking $\mathbb{E}[\,\cdot \mid \mathcal{F}_k]$ on (15), using $\mathbb{E}[\hat{g}_k \mid \mathcal{F}_k] = \nabla J(z_k)$ and (16), and using $1 - L\eta_k/2 \geq 1/2$ for $\eta_k \leq 1/L$:

$$\begin{aligned} \mathbb{E}[J(z_{k+1}) \mid \mathcal{F}_k] \;\geq\; & J(z_k) + \tfrac{\eta_k}{2}\,\|\nabla J(z_k)\|^2 \\ & - \tfrac{L\eta_k^2\sigma^2}{2}. \end{aligned} \tag{17}$$

Step 2 (telescoping). Take total expectation of (17), rearrange, sum from $k = 0$ to $K - 1$, and use (A3) ($\mathbb{E}[J(z_K)] \leq J^\star$):

$$\begin{aligned} \sum_{k=0}^{K-1} \eta_k\, \mathbb{E}[\|\nabla J(z_k)\|^2] \;\leq\; & 2(J^\star - J(z_0)) \\ & + L\sigma^2 \sum_{k=0}^{K-1} \eta_k^2. \end{aligned} \tag{18}$$

Step 3 (Part 1). Under (A4), $\eta_k > 0$, so each summand in (18) is non-negative and the partial sums are non-decreasing. Letting $K \to \infty$ and applying monotone convergence for non-negative series, together with $\sum_k \eta_k^2 < \infty$, gives $\sum_{k=0}^{\infty} \eta_k\, \mathbb{E}[\|\nabla J(z_k)\|^2] \leq M < \infty$ for the finite constant $M := 2(J^\star - J(z_0)) + L\sigma^2 \sum_{k=0}^{\infty} \eta_k^2$. If $\liminf_k \mathbb{E}[\|\nabla J(z_k)\|^2] > 0$, there exist $\varepsilon > 0$ and k_0 such that $\mathbb{E}[\|\nabla J(z_k)\|^2] \geq \varepsilon$ for all $k \geq k_0$ (covering both finite-positive and $+\infty$ cases), so $\sum_{k\geq k_0} \eta_k\, \mathbb{E}[\|\nabla J(z_k)\|^2] \geq \varepsilon \sum_{k\geq k_0} \eta_k = \infty$ by (A4), contradicting the bound. Hence $\liminf_k \mathbb{E}[\|\nabla J(z_k)\|^2] = 0$.

Step 4 (Part 2). With $\eta_k = \eta_0/\sqrt{k+1}$, $\eta_k > 0$ and $\eta_k \leq \eta_0 \leq 1/L$, so (18) applies. Standard integral comparisons give, for $K \geq 1$, $\sum_{k=0}^{K-1} 1/\sqrt{k+1} \geq 2(\sqrt{K}-1)$ and $\sum_{k=0}^{K-1} 1/(k+1) \leq 1 + \log K$, hence $\sum_k \eta_k \geq 2\eta_0(\sqrt{K} - 1)$ and $\sum_k \eta_k^2 \leq \eta_0^2(1 + \log K)$. Substituting into (18) and dividing by $\sum_k \eta_k > 0$ (for $K \geq 2$), combined with $\min_k \mathbb{E}[\|\nabla J(z_k)\|^2] \cdot \sum_k \eta_k \leq \sum_k \eta_k\, \mathbb{E}[\|\nabla J(z_k)\|^2]$, yields (14). □

Per-block step sizes. IntroLLM uses asymmetric per-block learning rates $\eta_{\theta,k}$ and $\eta_{\phi,k}$. Let $\eta_{\min,k} = \min(\eta_{\theta,k}, \eta_{\phi,k})$ and $\eta_{\max,k} = \max(\eta_{\theta,k}, \eta_{\phi,k})$. Repeating Steps 1–2 of the proof above with the block-diagonal step matrix $H_k = \mathrm{diag}(\eta_{\theta,k} I, \eta_{\phi,k} I)$ (the smoothness step is controlled by $\eta_{\max,k}$ and the descent term by $\eta_{\min,k}$) gives

$$\begin{aligned} &\sum_{k=0}^{K-1} \eta_{\min,k}\, \mathbb{E}[\|\nabla J(z_k)\|^2] \\ &\leq\; 2(J^\star - J(z_0)) + L\sigma^2 \sum_{k=0}^{K-1} \eta_{\max,k}^2, \end{aligned} \tag{19}$$

valid whenever $\eta_{\max,k} \leq 1/L$ and $\eta_{\min,k} > 0$ for every k. Two corollaries:

- *(a)* If $\sum_k \eta_{\min,k} = \infty$ and $\sum_k \eta_{\max,k}^2 < \infty$, the contradiction argument of Part 1 carries over to give $\liminf_k \mathbb{E}[\|\nabla J(z_k)\|^2] = 0$.
- *(b)* If $0 < c_{\min} \leq c_{\max} \leq 1/L$ and $\eta_{\min,k} =$

$c_{\min}/\sqrt{k+1}$, $\eta_{\max,k} = c_{\max}/\sqrt{k+1}$, then

$$\min_k \mathbb{E}[\|\nabla J(z_k)\|^2] \leq \frac{2(J^\star - J(z_0)) + L\sigma^2 c_{\max}^2(1+\log K)}{2\, c_{\min}(\sqrt{K}-1)},$$

which is $O((\log K)/\sqrt{K})$ with constants depending on $1/c_{\min}$ and $c_{\max}^2/c_{\min}$ separately (not on the ratio $\kappa = c_{\max}/c_{\min}$ alone — e.g. $c_{\min}, c_{\max} \to 0$ jointly preserves κ but blows up the initial-gap term).

IntroLLM is trained in practice with *constant* per-block learning rates $\eta_\theta = 1\times10^{-6}$, $\eta_\phi = 5\times10^{-5}$, which fall outside (A4) and (b). For constant schedules, (19) gives

$$\min_{0\leq k<K} \mathbb{E}[\|\nabla J(z_k)\|^2] \leq \frac{2(J^\star - J(z_0))}{K\,\eta_{\min}} + \frac{L\sigma^2\eta_{\max}^2}{\eta_{\min}},$$

so as $K \to \infty$ the best-iterate squared gradient norm is bounded by the irreducible noise floor $L\sigma^2\eta_{\max}^2/\eta_{\min}$. This is the standard constant-step-SGD floor for non-convex objectives — a bound on the best/average iterate, *not* convergence of the iterate sequence and *not* containment of all z_k in that neighbourhood. If the same constants are used as coefficients in a shared $1/\sqrt{k+1}$ schedule ($c_{\min} = 10^{-6}$, $c_{\max} = 5\times10^{-5}$, $\kappa = 50$), corollary (b) applies and the $O((\log K)/\sqrt{K})$ best-iterate rate is recovered.

PPO clipping as a bias-induced subsequence floor. The GRPO clipped surrogate replaces $\hat{g}_k$ by $\hat{g}_k^{\text{clip}} = \hat{g}_k - \delta_k$, where δ_k depends on the rollout at step k (hence is $\mathcal{F}_{k+1}$- but not $\mathcal{F}_k$-measurable). We add:

(A5) *Bounded clipping bias and variance.* δ_k is $\mathcal{F}_{k+1}$-measurable; its conditional mean $b_k := \mathbb{E}[\delta_k \mid \mathcal{F}_k]$ satisfies $\|b_k\| \leq c\,\epsilon$; and the conditional variance of the clipped estimator satisfies $\mathbb{E}[\|\hat{g}_k^{\text{clip}} - \mathbb{E}[\hat{g}_k^{\text{clip}} \mid \mathcal{F}_k]\|^2 \mid \mathcal{F}_k] \leq \sigma_{\text{clip}}^2$ for some $\sigma_{\text{clip}}^2 \geq \sigma^2$.

Bounded log-policy gradients give $\|b_k\| \leq c\,\epsilon$. Replacing $\hat{g}_k$ by $\hat{g}_k^{\text{clip}}$ in (15) and applying Young's inequality on the bias cross-term $\langle \nabla J(z_k), b_k\rangle$ yields, for $\eta_k \leq 1/(2L)$,

$$\begin{aligned}\mathbb{E}_k[J(z_{k+1})] \;\geq\; & J(z_k) + \tfrac{\eta_k}{4}\|\nabla J(z_k)\|^2 \\ & - \tfrac{L\eta_k^2\sigma_{\text{clip}}^2}{2} - \tfrac{\eta_k c^2\epsilon^2}{2} \\ & - L\eta_k^2 c^2\epsilon^2.\end{aligned}$$

Under the analogue of (A4) with cap $1/(2L)$, telescoping and dividing by $\sum_k \eta_k$ gives $\liminf_{k\to\infty} \mathbb{E}[\|\nabla J(z_k)\|^2] \leq 2c^2\epsilon^2$: a subsequence of expected squared-gradient values is asymptotically bounded above by $2c^2\epsilon^2$. This is *not* iterate convergence and *not* containment of all z_k in a neighbourhood, but a subsequence floor. The $O((\log K)/\sqrt{K})$ best-iterate rate to this floor is preserved under the schedule of Part 2 (with σ^2 replaced by σ_{clip}^2). With the default GRPO clip $\epsilon = 0.2$ and c of order unity, the floor is small in practice but irreducible under fixed-clip PPO.

Shared advantage and coupled noise. Theorem B.1 is stated conditional on (A2); we do not claim (A2) follows automatically from the bounded-reward setting. The exact GRPO advantage estimator $A_i = (R_i - \bar{R})/\hat{\sigma}$ with $\bar{R} = \frac{1}{G}\sum_j R_j$ admits three distinct variants:

- *Unnormalised centered*, $A_i^{\text{un}} = R_i - \bar{R}$: $\mathbb{E}[A_i^{\text{un}}\nabla\log\pi(\tau_i) \mid \mathcal{F}_k] = (1-1/G)\,\nabla J(z_k)$, a uniform multiplicative bias that is absorbed into the effective step size $\eta_k \leftarrow (1-1/G)\eta_k$ without affecting Theorem B.1.
- *Leave-one-out centered*, $A_i^{\text{loo}} = R_i - \frac{1}{G-1}\sum_{j\neq i} R_j$: $\mathbb{E}[A_i^{\text{loo}}\nabla\log\pi(\tau_i) \mid \mathcal{F}_k] = \nabla J(z_k)$ exactly.
- *Normalised GRPO*, $A_i = A_i^{\text{un}}/\hat{\sigma}$: the random denominator $\hat{\sigma}$ is correlated with the score $\nabla\log\pi(\tau_i)$, so the simple $(1-1/G)$ identity does not extend; (A2) holds in the regime where $\hat{\sigma}$ concentrates but is not exact at finite G.

With binary reward $R_i \in \{0,1\}$ and small G, the event $\hat{\sigma} = 0$ has positive probability (all G rewards equal); implementations resolve this with an additive stabiliser $\hat{\sigma}_\varepsilon = \sqrt{\widehat{\text{Var}} + \varepsilon}$, which we tacitly assume. The cross-block noise between $\hat{g}_{\theta,k}$ and $\hat{g}_{\phi,k}$ from the shared rollouts affects only the constant σ^2 in (A2), not the rate. A complete derivation of (A2) from any of the variants above also requires the standard score-function regularity conditions (interchange of ∇ and $\mathbb{E}$, bounded second moments of $\nabla\log\pi$, support stability), which we do not reproduce here.

C Implementation Details

C.1 Model Architecture

The model extends the base architecture with a lightweight MLP head branching from the final hidden state to predict temperature parameters.

Temperature Policy Head:

- **Structure**: Linear(d_{model}, $d_{\text{model}}/2$) $\rightarrow$ ReLU $\rightarrow$ Linear($d_{\text{model}}/2$, 3).
- **Output**: 3 logits $[u_c, u_\alpha, u_\beta]$ determining the temperature update probability and Beta distribution parameters.
- **Initialization**: Weights are initialized with $\mathcal{N}(0, 0.02)$. To facilitate initial exploration, final layer biases are set to $u_c = 0$ (implies $P(\text{change}) = 0.5$) and $u_\alpha = u_\beta \approx 0.541$ (implies Beta$(1, 1)$, i.e., Uniform distribution). This maximum-entropy initialization allows the model to explore the full temperature range early in training.

C.2 Hyperparameters

We list the detailed hyperparameters for the *IntroLLM* training in Table 4. To stabilize the joint optimization, we employ decoupled learning rates, setting the token policy to 1×10^{-6} and the temperature policy to 5×10^{-5}. The training process was executed on a cluster of 8 NVIDIA H200 GPUs using the AdamW optimizer, utilizing a global batch size of 128. We further set the group size to $G = 8$ to provide reliable relative advantage estimation during the rollout phase, while the maximum sequence length was capped at 3072 tokens to balance reasoning depth and compute efficiency.

Table 4: Hyperparameters for IntroLLM Training

Parameter	Value
Token Policy LR	1×10^{-6}
Temperature Policy LR	5×10^{-5}
Batch Size	128
Group Size (G)	8
Temperature Bounds $[\tau_{\min}, \tau_{\max}]$	$[0.6, 1.5]$
Max Sequence Length	3072
Optimizer	AdamW

C.3 Reward Function

We strictly follow the RLVR setting. The reward is binary: $R(x, y) = 1$ if the final answer matches the ground truth after symbolic verification, and $R(x, y) = 0$ otherwise. No partial rewards or process supervision rewards were used.

D Datasets

D.1 Training Set: MATH

We utilize the training split of the **MATH** dataset (Hendrycks et al., 2021), which contains 7,500 mathematics problems across seven categories (e.g., Algebra, Calculus, Counting & Probability). Each problem is accompanied by a step-by-step solution. Following standard RLVR practices, we only use the problem prompts for reinforcement learning, while the ground-truth answers serve as the basis for the verifiable reward function.

D.2 Evaluation Benchmarks

MATH-500 (Lightman et al., 2023) is a high-quality subset of the original MATH test set, sampled to provide a balanced representation of difficulty levels (L1–L5) and subject categories. It was first introduced by OpenAI to evaluate process-based reward models (PRMs). Given its manageable size and rigorous difficulty distribution, it serves as our primary benchmark for analyzing per-token temperature fluctuations and reasoning uncertainty.

AIME 2024 represents the most recent iteration of the American Invitational Mathematics Examination, a premier competition for high-school students. It consists of 15 highly complex problems where the answer is an integer between 000 and 999. Unlike standard datasets, AIME requires long-context logical deduction and is less susceptible to memorization, making it an ideal testbed for evaluating the exploration capabilities of IntroLLM on "out-of-distribution" hard problems.

AMC 2023 encompasses problems from the American Mathematics Competitions (AMC 10 and 12). These 25-question, 75-minute multiple-choice examinations are designed to challenge problem-solving skills under time constraints. We use the 2023 version to ensure that the models are evaluated on recent, potentially unobserved data. This benchmark tests the model's ability to maintain high precision while navigating the breadth of secondary school mathematics.

Minerva Math (Lewkowycz et al., 2022) is a specialized evaluation suite derived from the Minerva model's assessment framework. It primarily focuses on STEM-focused problem solving, drawing from sources like the MATH dataset and various university-level exams. We utilize this benchmark to evaluate the model's robustness in formal scientific reasoning and its ability to handle multi-step calculations that require consistent numerical accuracy.

OlympiadBench (He et al., 2024) is a comprehensive, bilingual (English and Chinese) benchmark featuring high-level competition problems

from International Mathematical Olympiads (IMO) and other prestigious contests. It includes both mathematics and physics domains. We leverage its mathematical portion to test whether the learned temperature policy can adapt to the extreme reasoning depths required for Olympiad-level deduction.

Omni-Math (Gao et al., 2024) is a recently released, large-scale benchmark designed to address the "saturation" of previous math datasets. It contains over 4,000 problems meticulously categorized into fine-grained sub-fields and difficulty tiers. Due to its vast coverage and high average difficulty, Omni-Math provides the most statistically significant measure of our model's general mathematical proficiency across the entire landscape of modern mathematics.

E Baseline Methods Details

To ensure the reproducibility of our comparative study, we provide the comprehensive technical configurations and hyperparameter settings for the two primary adaptive baselines. Both methods were integrated into our GRPO framework using the official logic described in their respective original works.

E.1 Standard GRPO (Fixed Temperature)

We evaluate three variants of the standard **GRPO** (Shao et al., 2024) using fixed sampling temperatures to establish baseline performance bounds for static exploration:

- **GRPO** ($\tau = 0.6$): Represents an **exploitative** configuration, focusing on high-probability reasoning paths to ensure output stability.
- **GRPO** ($\tau = 1.0$): The default setting used in most RLVR literature, providing a neutral balance between exploration and exploitation.
- **GRPO** ($\tau = 1.2$): Represents an **exploratory** configuration, intentionally increasing entropy to discover diverse reasoning trajectories.

In these baselines, the temperature remains constant throughout both the training and rollout phases, and across all token positions within a sequence.

E.2 EAD: Exploratory Annealed Decoding

EAD (Yang et al., 2025) focuses on *intra-sequence exploration*, grounded in the intuition that early tokens in a reasoning chain dictate the semantic direction and benefit most from high-entropy exploration, while later tokens should converge to stable exploitation.

Mechanism: The sampling temperature τ_t at token position t is governed by a dynamic simulated annealing schedule:

$$\tau_t = \max(\tau_{\min}, \tau_{\text{start}} \cdot \gamma^t),$$

where $\tau_{\text{start}} = 1.2$ and $\tau_{\min}$ is set to 0.1 for our 1.7B/4B models to encourage sharp convergence. To avoid disrupting the generation of standard response templates (e.g., "Let's think step by step"), we implement a **cutoff threshold** where τ_t remains fixed at τ_{start} for the first 10 tokens.

Global Adaptation: To account for the increasing length of reasoning chains during RL training, the decay rate γ is dynamically adjusted relative to the current training step n:

$$\gamma_n = \exp\left(-\frac{c_0}{L_{\text{avg}}} \cdot \frac{n}{N_{\text{total}}}\right),$$

where L_{avg} is the moving average response length and $c_0 = 25$ is the initial decay coefficient. This ensures that the annealing effect scales proportionally with the model's evolving verbosity. We also employ Truncated Importance Sampling (TIS) to maintain stability under high-variance off-policy updates.

E.3 TAMPO: Temperature Adaptive Meta Policy Optimization

TAMPO (Dang et al., 2026) treats temperature selection as a *meta-policy learning* problem, optimizing a discrete distribution over potential temperatures across training iterations.

Mechanism: TAMPO utilizes a hierarchical two-loop optimization process. In the **Inner Loop**, the model samples a temperature τ_{selected} from the current meta-policy to generate trajectories. In the **Outer Loop**, the meta-policy is updated based on **Temperature-Specific Advantages (TSAs)**. For a rollout y with reward R, the TSA for a candidate temperature τ_j is calculated as:

$$\text{TSA}(\tau_j) = A \cdot \frac{P(y \mid \tau_j)}{\sum_k P(y \mid \tau_k)},$$

where A is the group-relative advantage. This weighting reinforces temperatures that assign higher likelihood to high-reward trajectories.

Implementation Details:

- **Action Space:** A discrete set of 10 candidate values $\mathcal{T} = \{0.6, 0.7, \ldots, 1.5\}$.
- **Selection Strategy:** We apply Nucleus Sampling ($p = 0.9$) over the meta-policy distribution to select τ for each batch, ensuring a balance between exploiting the best temperature and maintaining meta-exploration.
- **Warmup & Stability:** The first 10% of training steps are designated as a warmup phase where τ is fixed at 1.0. We use an Exponential Moving Average (EMA) with $\alpha = 0.9$ to smooth the TSA updates and stabilize the meta-policy evolution.

E.4 Summary of Differences and Advantages

The fundamental distinction between IntroLLM and the aforementioned baselines lies in the granularity of control and the source of adaptation. We summarize these key differences in Table 5 and highlight our primary advantages below.

1. Introspective vs. Context-blind. Both Standard GRPO and TAMPO apply a single temperature to an entire response, ignoring the fact that reasoning uncertainty fluctuates within a chain; our ablation (Figure 7) shows that this sequence-level constraint alone costs 2.81%/1.38% average Avg@8 versus token-level control. TAMPO's meta-policy is moreover *unconditional*: at each step it samples from a $K{=}10$ categorical whose probabilities are computed entirely from batch-aggregated advantages, with no input signal from the prompt, prefix, or hidden state. EAD does adjust per token but follows a rigid hand-crafted decay schedule. In contrast, IntroLLM is **introspective**: π_ϕ conditions on h_t to detect when the model is entering a high-uncertainty reasoning pivot, allowing it to "look inward" to decide how much to "explore outward."

2. Reward-driven Emergence vs. Heuristic Design. EAD's annealing schedule and TAMPO's discrete bins rely on human intuition and manual tuning. TAMPO further smooths its meta-policy through EMA followed by min–max renormalisation, so the distribution behaves more like a slowly drifting annealing schedule than a reactive policy. IntroLLM treats temperature selection as a first-class control problem optimised directly by the GRPO clipped objective, so π_ϕ assimilates each batch's signal in a single PPO step. This allows the "reasoning rhythm" (Figure 4) to emerge naturally from the data, ensuring that exploration effort is allocated where it most contributes to finding the correct solution.

3. Efficiency through Mixed Action Space. Unlike token-level baselines that change temperature at every step and inject massive variance (the always-update ablation in Figure 7), or coarse-grid baselines like TAMPO that quantise τ into 10 bins with step $\Delta{=}0.1$ (so the achievable resolution is bounded by $\Delta/2$ regardless of training), our mixed **Discrete–Continuous action space** (Bernoulli trigger + Beta sampling) is continuous over $[\tau_{\min}, \tau_{\max}]$ and lets the model learn *when* to stay stable and *when* to adapt. This provides a superior balance between exploration flexibility and training stability, evidenced by the smoother convergence in our training curves (Figure 6).

F Supplementary Experiments

F.1 Multi-Seed Robustness

To verify that the main-paper numbers are not driven by a favourable random seed, we reran IntroLLM and all three baselines (GRPO ($\tau{=}1.0$), EAD, TAMPO) three times on each of Qwen3-1.7B-Base and Qwen3-4B-Base with seeds $\{42, 123, 456\}$, using identical hyperparameters and the matched 4-epoch training schedule. Table 6 reports mean $\pm$ standard deviation across the three runs for both Avg@8 and Pass@8 on AIME 2024, AMC 2023, and MATH-500, together with their average.

IntroLLM has the highest mean on every (scale, metric, benchmark) cell of Table 6. On the Mean column, it improves over the strongest baseline by 1.84/2.45 points on Avg@8 and 1.90/4.38 points on Pass@8 at the 1.7B/4B scales (EAD is the strongest baseline on Avg@8 at both scales; TAMPO on Pass@8). Against GRPO ($\tau{=}1.0$) the ± 1 std intervals are non-overlapping on every benchmark for either metric (e.g. AIME 2024 Avg@8 at 1.7B: IntroLLM $[8.80, 10.08]$ vs GRPO $[6.51, 7.93]$; at 4B: $[18.30, 19.76]$ vs $[13.46, 15.16]$). Against the stronger adaptive baselines EAD and TAMPO, IntroLLM remains non-overlapping on the hardest benchmark AIME 2024 at both scales (e.g. $[18.30, 19.76]$ vs EAD's $[13.21, 14.85]$ and TAMPO's $[14.00, 15.90]$ at 4B) and on AMC 23 at 1.7B; on the most saturated benchmark MATH-500, where headroom is small (0.5–0.8 points between IntroLLM and EAD in the main table), the intervals do touch or marginally overlap with EAD. The improvement direction is

Table 5: Conceptual comparison of IntroLLM against baseline temperature control methods.

Feature	Fixed GRPO	EAD	TAMPO	IntroLLM (Ours)
Control Level	Sequence	Token	Sequence	**Token**
Adaptation Basis	Static	Heuristic	Training Iteration	**Internal State (h_t)**
Optimization	None	None	Meta-Policy	**Joint RL**
Learned?	No	No	Yes (Partial)	**Yes (End-to-End)**

Table 6: **Multi-seed comparison on Qwen3-1.7B-Base and Qwen3-4B-Base.** Mean $\pm$ std over three independent runs with seeds $\{42, 123, 456\}$, using identical hyperparameters and the matched 4-epoch training schedule.

Scale	Metric	Method	MATH-500	AMC23	AIME24	Mean
Qwen3-1.7B	Avg@8	GRPO (τ=1.0)	65.07$\pm$0.42	37.81$\pm$0.83	7.22$\pm$0.71	36.70$\pm$0.48
		EAD (1.2$\rightarrow$0.1)	66.55$\pm$0.40	37.18$\pm$0.88	6.89$\pm$0.78	36.87$\pm$0.49
		TAMPO (0.6–1.5)	65.02$\pm$0.55	34.71$\pm$1.02	7.63$\pm$0.92	35.79$\pm$0.62
		IntroLLM (Ours)	**67.38$\pm$0.38**	**39.32$\pm$0.76**	**9.44$\pm$0.64**	**38.71$\pm$0.41**
	Pass@8	GRPO (τ=1.0)	82.47$\pm$0.55	66.83$\pm$1.04	17.22$\pm$1.18	55.51$\pm$0.72
		EAD (1.2$\rightarrow$0.1)	84.95$\pm$0.58	69.17$\pm$1.10	22.78$\pm$1.25	58.97$\pm$0.79
		TAMPO (0.6–1.5)	83.92$\pm$0.72	69.50$\pm$1.28	22.50$\pm$1.42	58.64$\pm$0.95
		IntroLLM (Ours)	**85.13$\pm$0.48**	**71.92$\pm$0.92**	**25.56$\pm$1.05**	**60.87$\pm$0.63**
Qwen3-4B	Avg@8	GRPO (τ=1.0)	79.24$\pm$0.38	49.79$\pm$0.91	14.31$\pm$0.85	47.78$\pm$0.45
		EAD (1.2$\rightarrow$0.1)	79.62$\pm$0.35	51.65$\pm$0.85	14.03$\pm$0.82	48.43$\pm$0.55
		TAMPO (0.6–1.5)	78.10$\pm$0.48	50.58$\pm$0.98	14.95$\pm$0.95	47.88$\pm$0.65
		IntroLLM (Ours)	**80.91$\pm$0.32**	**52.71$\pm$0.78**	**19.03$\pm$0.73**	**50.88$\pm$0.38**
	Pass@8	GRPO (τ=1.0)	91.83$\pm$0.48	79.58$\pm$1.08	33.61$\pm$1.28	68.34$\pm$0.69
		EAD (1.2$\rightarrow$0.1)	91.93$\pm$0.45	77.32$\pm$1.05	36.39$\pm$1.35	68.55$\pm$0.80
		TAMPO (0.6–1.5)	91.58$\pm$0.58	82.16$\pm$1.22	33.05$\pm$1.45	68.93$\pm$0.92
		IntroLLM (Ours)	**93.14$\pm$0.42**	**87.08$\pm$0.95**	**39.72$\pm$1.10**	**73.31$\pm$0.55**

consistent across all 24 (method-pair $\times$ benchmark $\times$ metric $\times$ scale) comparisons, indicating that the gain is robust to seed choice rather than a single-run artefact.

F.2 Interpretable Adaptation Across Difficulty and Semantic Role

Beyond the training-time trajectory shown in the main paper (Figure 5), we examine two complementary slices of the learned temperature policy: how it scales with problem difficulty, and which token types it tends to assign high temperatures to.

Difficulty-Aware Temperature Scaling. Figure 8 plots the distribution of predicted temperatures across MATH-500 difficulty levels (L1–L5). Median temperature increases monotonically from L1 to L5, demonstrating a clear positive correlation between problem hardness and exploration intensity. The policy autonomously reads difficulty from the model's hidden state and allocates larger exploration budgets to harder problems—a key driver of the gains on challenging benchmarks like AIME and Omni-Math.

Targeted Exploration at Reasoning Keywords. To identify the semantic patterns behind temperature spikes, we collect the top-100 highest-temperature tokens across all MATH-500 trajectories. Figure 9 shows that IntroLLM concentrates high temperatures on reasoning-critical keywords such as "assume", "consider", "finding", and "different". This aligns with recent findings that a small set of critical tokens drives exploration in reasoning chains (Wang et al., 2025c), and shows that reward-driven training automatically discovers this structure without manual annotation.

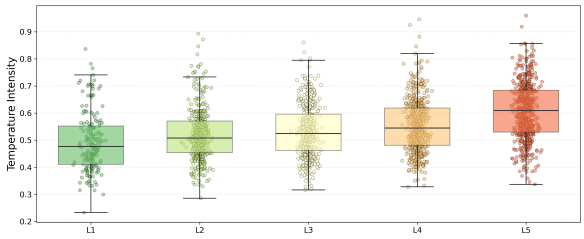

Figure 8: **Distribution of predicted temperatures across MATH-500 difficulty levels (L1–L5, from easy to hard).** As problems get harder, the average temperature tends to increase.

F.3 Full Ablation Numbers

Table 7 reports the full ablation underlying Figure 7, including AMC23 and the per-scale averages, in

Table 7: **Ablation on temperature strategies.** We report Avg@8 (A@8) and Pass@8 (P@8) in %.

Method	Qwen-1.7B-Base								Qwen-4B-Base							
	AIME 24		AMC23		MATH		Average		AIME 24		AMC23		MATH		Average	
	A@8	P@8	A@8	P@8	A@8	P@8	A@8	P@8	A@8	P@8	A@8	P@8	A@8	P@8	A@8	P@8
Prompt-level	7.08	20.00	35.31	67.50	65.35	85.40	35.91	57.63	17.92	33.33	49.38	82.50	**80.97**	92.40	49.42	69.41
Always-update	7.08	23.33	**39.69**	67.50	66.75	84.80	37.84	58.54	16.25	36.67	47.19	85.00	80.17	91.80	47.87	71.16
IntroLLM	**9.58**	**26.67**	39.06	**72.50**	**67.53**	**86.20**	**38.72**	**61.79**	**19.17**	**40.00**	**52.50**	**87.50**	80.73	**93.00**	**50.80**	**73.50**

Figure 9: **Reasoning keywords trigger high temperatures.** Top 100 highest-temperature tokens across MATH-500. Keywords like "assume", "consider", and "finding" consistently receive increased exploration.

Table 8: **Training-time wall-clock overhead of IntroLLM across model scales.** All runs use 4 training epochs on 8×H200 with identical training settings.

Scale	GRPO	IntroLLM	Δ Time
1.7B	2h 11m 58s	2h 32m 54s	+20m 56s
4B	3h 14m 29s	3h 40m 47s	+26m 18s
14B	11h 41m 17s	12h 23m 54s	+42m 37s

both Avg@8 (A@8) and Pass@8 (P@8) formats.

F.4 Training-Time Overhead Across Model Scales

The efficiency analysis in the main paper (Qwen3-1.7B) shows that the temperature head adds only 0.122% parameters and a small inference-time cost. Here we extend this analysis to training-time wall-clock on three model scales (1.7B, 4B, 14B), comparing IntroLLM against the GRPO baseline under matched 4-epoch training runs on 8×H200 GPUs.

Table 8 reports the wall-clock measurements. The absolute extra time introduced by IntroLLM grows mildly with model size (from 21 minutes at 1.7B to 43 minutes at 14B), but the *relative* overhead does not worsen, indicating that IntroLLM remains practical as the base model is scaled up.

Two architectural factors explain this favourable scaling behaviour. *(i) Compute complexity vs. backbone scaling.* The added module is a lightweight two-layer MLP on the final hidden state plus the cost of sampling from the Bernoulli/Beta temperature policy. Its FLOPs and memory footprint are small compared with a full transformer forward/backward pass. *(ii) Overhead dilution as models grow.* The temperature head scales with a single low-dimensional projection, while the backbone processes all tokens through every transformer layer; as model size grows, the backbone's cost dominates and the controller's relative cost is diluted. Together these properties suggest that IntroLLM remains practical at larger scales: the overhead/performance trade-off does not deteriorate as the base model is scaled up.

G Declaration of LLM Usage

We declare that Large Language Models were utilized exclusively for grammatical refinement and spell-checking to enhance the clarity and readability of this manuscript. No part of the intellectual content, experimental design, or results was generated by AI.