LET IT CALM: EXPLORATORY ANNEALED DECODING FOR VERIFIABLE RL IN MATHEMATICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) is a powerful paradigm for enhancing the reasoning capabilities of large language models (LLMs), yet its success hinges on effective exploration. An ideal exploration strategy must navigate two fundamental challenges: it must preserve sample quality while also ensuring training stability. While standard fixed-temperature sampling is simple, it struggles to balance these competing demands, as high temperatures degrade sample quality and low temperatures limit discovery. In this work, we propose a simpler and more effective strategy, Exploratory Annealed Decoding (EAD), grounded in the insight that exploration is most impactful on early tokens which define a sequence's semantic direction. EAD implements an intuitive *explore at the beginning, exploit at the end* strategy by annealing the sampling temperature from high to low during generation. This dynamic schedule encourages meaningful, high-level diversity at the start, then gradually lowers the temperature to preserve sample quality and keep the sampling distribution close to the target policy, which is essential for stable training. We demonstrate that EAD is a lightweight, plug-and-play method that significantly improves sample efficiency, consistently outperforming fixed-temperature sampling across various RLVR algorithms and model sizes on mathematical reasoning tasks. Our work suggests that aligning exploration with the natural dynamics of sequential generation offers a robust path to improving LLM reasoning.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) is a powerful approach to enhance the capabilities of Large Language Models (LLMs), particularly in mathematical reasoning (OpenAI, 2024; Guo et al., 2025; Team et al., 2025; Yang et al., 2025). In this framework, an LLM learns by iteratively generating potential solutions (i.e., rollouts), and receiving feedback on its attempts. A central challenge lies in guiding language models to explore diverse yet high-quality solutions in their vast output space (Cheng et al., 2025); this reflects the long-standing hard trade-off between exploration and exploitation in RL (Thrun, 1992; Sutton et al., 1998).

To achieve effective exploration, the sampling process can be modified to increase the variance of its underlying distribution. However, any such modification involves two fundamental challenges.

First, it must **preserve sample quality**. Increasing diversity at the cost of generating low-quality, nonsensical outputs is counterproductive. Second, it must **ensure training stability**. Modifying the sampler creates a discrepancy between the behavior policy (used for sampling) and the target policy (being optimized), necessitating an importance sampling (IS) correction in the gradient update (Degris et al., 2012). Large IS probability ratios can cause high-variance gradients that destabilize the entire training process (Schulman et al., 2017). An ideal exploration technique must therefore increase diversity while keeping the sampling distribution close enough to the target policy to allow for stable learning (Haarnoja et al., 2018; Ziegler et al., 2019).

A principled approach to this trade-off is adjusting the sampling temperature (Ackley et al., 1985; Hou et al., 2025), which is both simple and *variationally optimal* (i.e., maximizing entropy and thereby increasing diversity while bounding KL divergence from the target policy (Jaynes, 1957)). However, relying on a single fixed temperature creates a tension: high temperature promotes diversity but produces nonsensical text (Renze, 2024; Wang et al., 2025b), whereas low temperature improves

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.

quality but limits exploration, yielding generic and repetitive outputs (Holtzman et al., 2020; Guo et al., 2025).

In this work, we study this trade-off in *math* RLVR, where a few early reasoning tokens typically commit the model to a solution path. We propose **Exploratory Annealed Decoding (EAD)**, which improves the balance by leveraging a key insight into sequential generation: exploration is not equally valuable at every step. The initial tokens shape a sequence's semantic direction and structure, making early exploration crucial for discovering diverse valid solutions. Later tokens, however, fill in details within the established context, where excessive exploration can harm coherence. This insight motivates our core strategy: *explore at the beginning, exploit at the end*. This simple principle elegantly addresses the twin challenges of quality and stability. Injecting randomness early promotes diverse, high-level exploration, while reducing it later ensures completions are both coherent and close to the target policy—an essential property for stable off-policy learning. Extensions beyond math (e.g., code and agentic tasks) are discussed in § 7.

In summary, our contributions are as follows:

① We propose EAD, a simple and effective exploration strategy for RLVR that dynamically anneals temperature to encourage diversity while maintaining high sample quality.
② We show EAD is a plug-and-play enhancement that improves sample efficiency over temperature sampling, delivering robust gains across various RLVR algorithms including GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), and EntropyMech (Cui et al., 2025) on both small and larger models.
③ We show that EAD can be adapted for test-time inference, where a tuned temperature schedule further enhances generation quality.

2 Preliminary

Notations. Let x be a prompt from a dataset $\mathcal{D}$, and let $y = (y_1, \ldots, y_{|y|})$ be a generated response sequence, where $y_{<t}$ denotes the prefix $(y_1, \ldots, y_{t-1})$. We define an LLM as a policy π_θ parameterized by θ, and denote the reference policy, i.e., the starting point for RL, as π_{ref}. The probability of generating y given x is defined autoregressively as $\pi_\theta(y \mid x) = \prod_{t=1}^{|y|} \pi_\theta(y_t \mid [x, y_{<t}])$. A reward model $R(x, y)$ evaluates the quality of a prompt-response pair; for our RLVR experiments, we use the rule-based Math-Verify reward model.[1] Finally, we use $|\cdot|$ to denote the length of a sequence or the cardinality of a set, and use the shorthand $1:n$ for the set $\{1, \ldots, n\}$.

Reinforcement Learning with Verifiable Rewards (RLVR). The standard objective for reinforcement fine-tuning of LLMs is to maximize the expected reward over a prompt dataset $\mathcal{D}$:

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

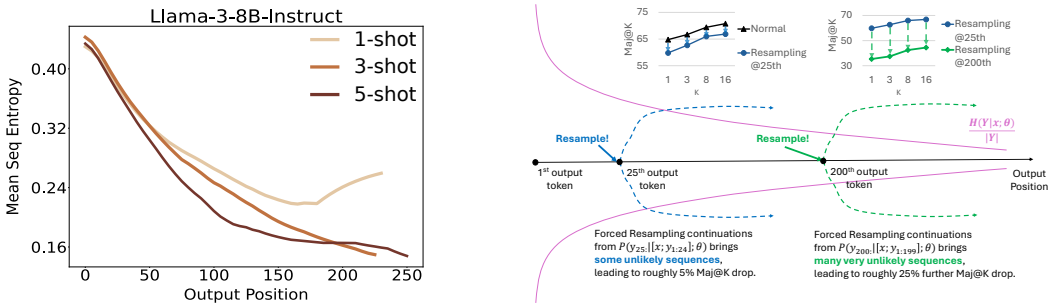
However, on complex reasoning tasks, learned reward models $R(x, y)$ are prone to *reward hacking*, where they assign high scores to plausible but incorrect solutions (Gao et al., 2023; Perez et al., 2023; Weng, 2024; Wang et al., 2025a). RLVR addresses this by replacing the learned model with a verifiable, rule-based reward signal—such as a verifier that provides binary feedback on a solution's correctness (Guo et al., 2025). This ensures that the policy is optimized using a reliable signal.

RLVR is commonly implemented using policy gradient algorithms like Proximal Policy Optimization (PPO) (Schulman et al., 2017). However, standard PPO often requires complex token-level advantage estimation and a separate value model. To better suit RLVR's trajectory-level binary rewards, subsequent methods simplify the advantage calculation (Shao et al., 2024; Yu et al., 2025). A prominent example is Decoupled Clip and Dynamic Sampling Policy Optimization (DAPO) (Yu et al., 2025), which optimizes:

$$J_{\text{DAPO}}(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D}, \\ y^{(1:G)} \overset{iid}{\sim} \pi_{\theta_{\text{old}}}(\cdot|x)}} \left[\frac{1}{\sum_{i=1}^{G} |y^{(i)}|} \sum_{i=1}^{G} \sum_{t=1}^{|y^{(i)}|} \min \left\{ r_t^{(i)}(\theta) A_i, \text{clip} \left(r_t^{(i)}(\theta), 1-\varepsilon_{\text{low}}, 1+\varepsilon_{\text{high}} \right) A_i \right\} \right]$$

$$\text{s.t.} \quad 0 < \left| \{ y^{(i)} : y^{(i)} \text{ is correct} \} \right| < G,$$

[1] https://github.com/huggingface/Math-Verify



(a) (b)

Figure 1: (a) Avg. entropy shrinks with output positions for Llama-3-8B-Instruct on MMLU dataset; (b) Forking on DeepSeek-Llama-3: resample a new suffix from a frozen prefix at token 25 vs. 200; early forks (high-entropy) keep Maj@k, late forks (low-entropy) do not.

where $\pi_{\theta_{\text{old}}}$ refers to previous policy and $r_t^{(i)}(\theta) = \frac{\pi_\theta(y_t^{(i)}|[x,y_{<t}^{(i)}])}{\pi_{\theta_{\text{old}}}(y_t^{(i)}|[x,y_{<t}^{(i)}])}$. The asymmetric bound $\varepsilon_{\text{high}} > \varepsilon_{\text{low}}$ is proposed to relax the restriction on probability increase and encourage exploration in the training. The advantage A_i is computed by normalizing the binary rewards across a batch of G responses, thus avoiding the need for a value model:

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

Temperature. Temperature sampling (Ackley et al., 1985) is a widely used method to control the stochasticity of the policy π_θ. At each generation step t, the LLM computes a vector of logits, $\mathbf{h}$, over the vocabulary V based on the prompt x and the preceding tokens $y_{<t}$. Temperature sampling rescales these logits with a parameter $\tau > 0$ before applying the softmax function to form the next-token probability distribution: $\pi_\theta(y_t = v \mid [x, y_{<t}]; \tau) = \frac{\exp(h_v/\tau)}{\sum_{v'\in V}\exp(h_{v'}/\tau)}$, where v is a token in the vocabulary V and h_v is its corresponding logit. The temperature τ directly modulates the sharpness of the output distribution. A higher temperature ($\tau > 1$) flattens the distribution, increasing output diversity by making less likely tokens more probable. Conversely, a lower temperature ($\tau < 1$) sharpens it, leading to more deterministic, greedy outputs.

Pass@k. Pass@k measures the probability that at least one of k independent outputs from a language model is correct. Let p_x denote the per-prompt accuracy of the model π. The pass@k accuracy is defined as $\mathbb{E}_{x\sim\mathcal{D}}\left[1-(1-p_x)^k\right]$. High diversity benefits this metric, as a model with diverse outputs is more likely to maintain non-negligible p_x across prompts.

Entropy. Given a prompt x and a prefix $y_{<t}$, the **token-level entropy** at step t is defined over the policy's conditional distribution:

$$H(Y_t \mid [x, y_{<t}]; \theta) := -\sum_{v\in V} \pi_\theta(y_t = v \mid [x, y_{<t}]) \log \pi_\theta(y_t = v \mid [x, y_{<t}]),$$

where the sum is over all tokens v in the vocabulary V.

The **average entropy** of the policy, $H(\pi_\theta)$, is the expected token-level entropy over all prompts and generation steps. We can estimate this value empirically using Monte Carlo sampling. For each prompt $x \in \mathcal{D}$, we generate G i.i.d. responses $y^{(1)}, \ldots, y^{(G)}$. The average entropy is then approximated by averaging the token-level entropies across all tokens:

$$\bar{H}(\pi_\theta) \approx \frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}}\left(\frac{1}{\sum_{i=1}^{G}|y^{(i)}|}\sum_{i=1}^{G}\sum_{t=1}^{|y^{(i)}|}H(Y_t \mid [x, y_{<t}^{(i)}]; \theta)\right). \tag{2}$$

3 Sequential Exploration: Explore Early, Exploit Late

Exploration is a cornerstone of reinforcement learning, enabling agents to discover high-quality policies rather than settling on suboptimal solutions (Sutton et al., 1998; Ladosz et al., 2022). This principle becomes particularly vital in deep RL, where vast action spaces render exhaustive search

infeasible. In the context of RL for language models (e.g., RLVR), insufficient exploration often manifests as *entropy collapse*, i.e., a premature narrowing of the generation distribution during training (Yu et al., 2025; Wang et al., 2025b; Cui et al., 2025).

A common simple tool to encourage exploration is *temperature sampling*. However, a *fixed* temperature imposes a difficult trade-off. A high temperature promotes diversity (as indicated by increased entropy[2]), but it risks degrading output quality with nonsensical tokens and hallucinations (Renze, 2024; Wang et al., 2025b). In contrast, a low temperature limits the discovery of novel solutions, leading to generic and repetitive outputs (Holtzman et al., 2020; Guo et al., 2025).

The key to resolving this dilemma lies not in finding a single best temperature, but in recognizing that exploration requirements vary throughout the generation process. This key insight stems directly from the autoregressive nature of language models. At the beginning of a sequence, the context is minimal and uncertainty is high, allowing a wide range of valid continuations. As more tokens are produced, the context becomes increasingly specific, constraining subsequent choices. Fig. 1a confirms this empirically: on MMLU (Hendrycks et al., 2021) with Llama-3-8B-Instruct (Grattafiori et al., 2024),[3] position-wise entropy declines as generation proceeds.

This entropy decay has a direct performance implication: effective exploration, producing diverse, high-quality responses, is most beneficial in the uncertain, high-entropy phase at the start of generation. To verify this, we replicate the mid-generation "forking" experiment of Yang et al. (2026) on DeepSeek-Distilled Llama-3-8B (Guo et al., 2025). After generating a response, we keep a prefix $y_{<t}$ and resample a new continuation from $P(Y_{\geq t} \mid [x, y_{<t}])$, scoring the completed sequence by majority vote (Maj@K). As shown in Fig. 1b, forking at the 25th token (still high-entropy) barely hurts Maj@K, whereas forking at the 200th token (low-entropy) drops it sharply: late branches leave an already-committed path and become low-quality. This finding is also consistent with observations from inference-time analysis, where forced, late-stage exploration tends to degrade output quality (Liao et al., 2025; Yang et al., 2026; Fu et al., 2025). The same position-wise narrowing is reminiscent of the Data Processing Inequality (Shannon, 1948).[4]

Following this empirical pattern, we arrive at a simple yet powerful design principle: **explore early and exploit late**. This is a practical heuristic grounded in the entropy dynamics of autoregressive models, not a claim that late-stage exploration is never useful. Individual sequences may contain critical late-stage decisions; however, in expectation, the marginal value of exploration is highest at the beginning. Moreover, EAD does not eliminate late-stage stochasticity entirely—the floor temperature $\tau_{\min} > 0$ ensures that a non-trivial degree of randomness is retained throughout generation.

4 A Method for Sequential Exploration

To put the principle of "explore early, exploit late" into practice, we introduce *Exploratory Annealed Decoding (EAD)*, which uses an annealed temperature schedule starting from a higher-than-standard initial temperature (i.e., $\tau > 1$). To adapt this strategy to RLVR, we further incorporate a *global-step-aware decay rate*, ensuring that the temperature schedule remains effective as the typical response length increases during training.

Exploratory Annealed Decoding. Instead of a fixed temperature, our method dynamically adjusts the temperature τ_t for each token t in a rollout. The schedule starts at a high temperature $\tau_{\max} > 1$ and decreases progressively throughout the generation process. Specifically, we sample the t-th token for one rollout with the token-level temperature $\tau_t = \max\{1 + \tau_{\max} - e^{t/d}, \tau_{\min}\}$, where we apply the annealed schedule with a *decay rate* d controlling the annealing speed.

[2] See Appendix F for the proof.

[3] We use MMLU as a held-out dataset with Chain-of-Thought prompting (Wei et al., 2022) to incentivize longer reasoning outputs, matching a typical RLVR scenario.

[4] The DPI bounds entropy along a Markov chain $X \to Y \to Z$ and does not formally govern the position-wise conditional entropy of a fixed-policy autoregressive model. We mention it only as informal intuition for why expected next-token uncertainty tends to fall as more context is revealed; the claim we rely on is the empirical entropy decay (Fig. 1a) and the forking result (Fig. 1b). Late high-entropy positions can still occur, but become exponentially rare with position t (Yang et al., 2026).

Algorithm 1 Exploratory Annealed Decoding

Require: prompt x; policy π_θ; hyperparameters $\tau_{\max}, \tau_{\min}, c, d_0, \alpha, d_{\max}$; global step s
$d_s \leftarrow \min(d_0 + \alpha s, d_{\max})$; $t \leftarrow 1$; $y \leftarrow ()$ {t: absolute output position}
while not finished **do**
  $\tau_t \leftarrow 1.0$ **if** $t < c$ **else** $\max\{1 + \tau_{\max} - e^{t/d_s}, \tau_{\min}\}$ {τ=1 on template prefix}
  sample $y_t \sim \pi_\theta(\cdot \mid [x, y_{<t}]; \tau_t)$; append y_t to y; $t \leftarrow t + 1$
end while
return response y

As illustrated in Fig. 2, the decay rate d controls how long the policy remains in a high-exploration state. A larger d front-loads exploration across more initial tokens, while a smaller d transitions to exploitation more quickly. We adopt the negative-exponential form because it allows a relatively slow decay at the very beginning of generation, where maintaining exploration is most valuable, while gradually encouraging more stable, higher-quality continuations later in the rollout. We do not claim that this specific functional form is uniquely optimal: EAD is best understood as a practical instance of *position-aware annealed rollout sampling*, and other monotonically decreasing schedules (e.g., linear or two-stage) may perform comparably with properly tuned hyperparameters. We discuss this scope and possible alternative schedules in Appendix I.

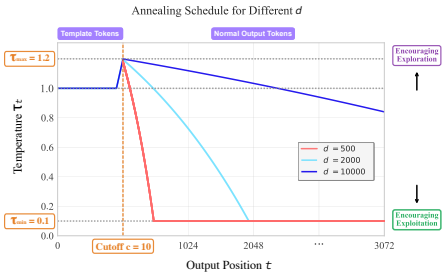


Figure 2: The annealing schedule with different decay rates d. A larger d slows the cooling, front-loading exploration. We set $c = 10, \tau_{\max} = 1.2, \tau_{\min} = 0.1$.

In practice, we let $\tau_t = 1.0$ for $t < c$, where c is a pre-determined initial position for the sake of model-specific or prompt-specific template tokens injected in the training process. During RLVR, language models tend to generate some template tokens such as "*let's verify step by steps*" or repeat the question. We fix the temperature at $\tau = 1.0$ in this part to avoid interfering with the generation process. Typically, c is very small (e.g., a few dozen tokens corresponding only to the prompt or template length) and does not affect the core reasoning steps where exploration is most critical. Throughout, the token index t denotes the *absolute* output position and is *not* reset after the cutoff c: the schedule $\tau_t = \max\{1 + \tau_{\max} - e^{t/d}, \tau_{\min}\}$ applies as-is for $t \geq c$. Because c is small (typically $c = 10$) and the schedule is smooth, the schedule value at $t = c$ is essentially $\tau_{\max}$, so the cutoff acts as a "delayed start" rather than a reset of the annealing curve.

Global-Step-Aware Decay Rate. As training progresses and response lengths increase,[5] the decay rate d should be adjusted in accordance with the training step. Otherwise, an excessive number of tokens may be generated under extremely low temperatures, which degrades response quality and leads to undesirable behaviors such as repetition (Guo et al., 2025), off-topic drift (Spataru et al., 2024), and unnecessary verbosity (Holtzman et al., 2020). In particular, we adopt the following *global-step-aware decay rate*: $d_s = \min(d_0 + \alpha \times s, d_{\max})$, where $\alpha > 0$ is the growth factor and $d_{\max}$ is the decay cap. Algorithm 1 summarizes the EAD rollout process.

Ensuring Stability with Truncated Importance Sampling. With aggressive annealing schedules (e.g., very small $\tau_{\min}$ and d), sampling low-probability, long-tail tokens can cause the annealed policy to deviate significantly from the one being optimized. This creates an off-policy discrepancy that risks training instability. To mitigate this, we employ *truncated importance sampling* (TIS) (Heckman et al., 1998; Hilton et al., 2022; Yao et al., 2025) to reduce the off-policy gap, ensuring stable optimization even under highly exploratory schedules. We emphasize that TIS is a *biased* variance-control mechanism rather than an unbiased correction: the sequence-level truncation trades a small downward bias on the high-ratio tail for a substantial reduction in gradient variance (see Appendix B for the exact form, practical choices, and bias–variance trade-off).

Overall, this annealed decoding strategy offers a compelling combination of effectiveness and efficiency. As a plug-and-play modification to standard temperature sampling, it incurs negligible computational overhead and is fully compatible with existing RLVR pipelines and diverse policy optimization algorithms like DAPO, GRPO, etc.

[5]We illustrate increased length for EAD in Appendix G.

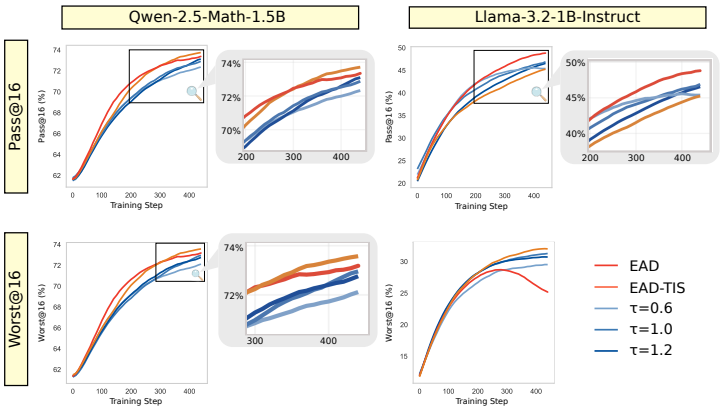


Figure 3: Pass@16 and Worst@16 evaluation in RL training. While EAD improves exploration of high-quality samples (even the worst outperform temperature sampling), the gain diminishes over time; TIS can supplement to correct bias and sustain training.

5 Experiments

5.1 Experimental Setup

Models, Data, and Training Frameworks. To ensure a rigorous and controlled comparison, we follow the Minimal-RL recipe (Xiong et al., 2025),[6] training all models on the Numina-Math dataset (Beeching et al., 2024), which contains 860k math prompts. We experiment with both Qwen-2.5-Math-1.5B (Yang et al., 2024) and Llama-3.2-1B-Instruct (Dubey et al., 2024).[7] We also include the larger Qwen-2.5-Math-7B model to evaluate how our approach scales. While our primary experiments are conducted within the DAPO framework (Yu et al., 2025), we demonstrate broader applicability by additionally integrating EAD with GRPO (Shao et al., 2024) and EntropyMech (Cui et al., 2025).

Baselines and Controlled Comparison. We evaluate EAD against fixed-temperature sampling, with temperatures $\tau \in \{0.6, 1.0, 1.2\}$ as recommended by prior work (Renze, 2024; Guo et al., 2025; Hou et al., 2025). For a fair comparison focused specifically on the sampling strategy, we disable two orthogonal techniques for all methods: (1) dynamic data sampling (Yu et al., 2025), to maintain a consistent training set, and (2) rollout length penalties, to avoid confounding the reward signal with length-based biases.

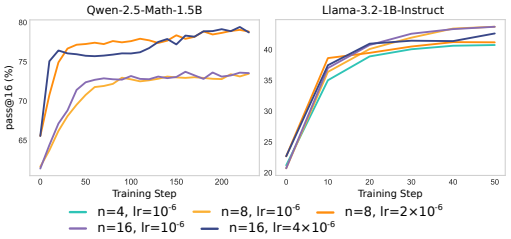
Hyperparameters and Tuning. Unless otherwise stated, EAD uses $\tau_{\max} = 1.2$, $d_0 = 25$, $\alpha = 5$, $d_{\max} = 40000$, and a model-dependent floor $\tau_{\min}$: 0.1 for the 1B/1.5B models and 0.8 for the more capable 7B model, whose sharper next-token distribution benefits from a higher floor that avoids overconfident, near-greedy decoding; the "explore early, exploit late" dynamic is preserved even within this narrower range. Learning rates for both EAD and the fixed-temperature baselines were tuned on held-out data for a fair comparison, and we ablate hyperparameter sensitivity in § 5.5.

5.2 EAD Improves RLVR Training

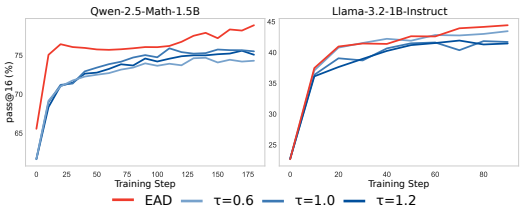
EAD Improves RL Exploration and Training Efficiency. As shown in Fig. 3, EAD significantly improves training efficiency. For Pass@16 accuracy, EAD (w/o TIS) consistently outperforms the baselines on the Llama and Qwen models (EAD (w/ TIS) also outperforms on the Llama model), demonstrating more effective exploration. Under the stricter Worst@16 metric, the inclusion of TIS becomes essential for maintaining stable performance gains, highlighting its importance for correcting the off-policy training dynamic introduced by EAD. Through bootstrapping evaluation as in Hochlehnert et al. (2025), the standard deviation of both Pass@16 performance and Worst@16 are way below 0.01 and thus all comparisons here are significant.

[6] More training details and hyperparameter setups are illustrated in Appendix A.

[7] We also experimented with the Llama-3.2-1B base model. However, consistent with Wang et al. (2025c), we found that applying RL to base models without intermediate domain-specific fine-tuning yields limited gains across all methods. We defer a deeper investigation to future work.



(a) EAD would bring further performance improvement via increased numbers of rollouts, but the commonly used 4 / 8 is already good enough.



(b) When scaling out the rollout number to 16, the relative advantages of EAD diminished; however, it still outperforms traditional same-temperature sampling.

Figure 7: Impact of rollout numbers for EAD.

To verify that these gains are not merely a multi-sample diversity effect, we also report final-checkpoint Pass@1 in Appendix J (Table 3): EAD improves single-sample accuracy over every fixed-temperature baseline. To verify that our method generalizes, we evaluated it on the larger Qwen-2.5-Math-7B model. The results, presented in Fig. 4, confirm that the performance gains from EAD remain significant. This demonstrates that our approach is effective not only on smaller models but also scales successfully.

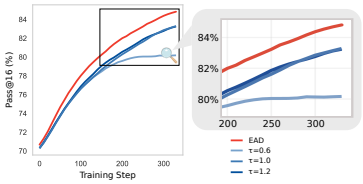


Figure 4: Pass@16 on Qwen-2.5-Math-7B. EAD yields sustained gains over fixed-temperature sampling.

EAD Mitigates Entropy Collapse. One major problem in RLVR training is entropy collapse (Cui et al., 2025), which shrinks the exploration space and constrains improvement during the "plateau stage" (Deng et al., 2025). As shown in Fig. 5, the entropy of EAD-empowered methods is not monotonically decreasing; instead it gradually transitions out of local optima (Kirkpatrick et al., 1983; Bertsimas & Tsitsiklis, 1993) in a natural, continuous way, without external intervention such as tree search in rollout sampling (Li et al., 2025).

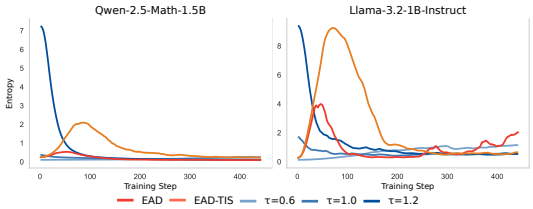


Figure 5: Entropy dynamics. Fixed temperature sampling leads to a sharp decrease, while EAD maintains exploration to escape local minima.

Sample efficiency of EAD. Increasing the number of rollouts enhances exploration but is computationally expensive (Hou et al., 2025). Varying the rollout count (adjusting the learning rate accordingly (Chen et al., 2025)), Fig. 7a shows that EAD already achieves strong results with just 4 or 8 rollouts (e.g., $n = 8$ with learning rate 10^{-6} for Llama-3.2-1B-Instruct and 2×10^{-6} for Qwen-2.5-Math-1.5B), reducing the cost of the rollout phase.

5.3 EAD is Compatible with Various RL Algorithms

To demonstrate that EAD is a general, plug-and-play strategy, we integrate it into two other RL algorithms with distinct dynamics: GRPO (Shao et al., 2024), which constrains updates with a KL penalty and stricter clipping that can limit exploration (Yu et al., 2025), and EntropyMech (Cui et al., 2025), which uses token-clipping to mitigate entropy collapse. As shown in Fig. 6, EAD consistently outperforms fixed-temperature sampling in both, confirming its broad applicability across RL algorithms as an improved exploration strategy.

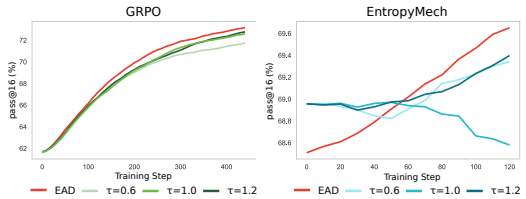


Figure 6: EAD compatibility across RL algorithms. It consistently improves model performance over time compared to baselines.

5.4 EAD Improves Inference-Time Scaling

To isolate the sampling strategy from RL optimization dynamics (Berseth, 2025), we evaluate EAD at inference time on off-the-shelf Qwen-2.5 models without fine-tuning. Since no oracle selects the correct answer among N candidates, we use majority voting (Majority@N) (Wang et al., 2023; Snell et al., 2024). As shown in Fig. 8, EAD consistently improves over the baseline for most values of N. This result suggests that EAD's temperature schedule produces samples that are not only diverse but also of sufficiently high quality to improve the majority-voted consensus, even without any RL training.

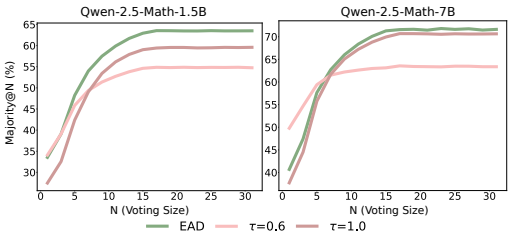


Figure 8: Inference-Time Scaling for Qwen2.5. EAD improves traditional sampling. Parameters: $\tau_{\max} = 1.2, \tau_{\min} = 0.1, d = 25$.

5.5 Ablation Study

Reverse Annealing. We also test a reverse schedule, dubbed "RevEAD", where temperature increases during generation: $\tau_t = \min\{\tau_{\min} + e^{t/d}, \tau_{\max}\}$. As shown in Fig. 9, RevEAD fails to even outperform standard temperature sampling, validating the necessity of an annealing (decreasing) schedule. This supports our hypothesis that late-stage exploration can be detrimental, yielding high-variance "lucky" successes rather than robust reasoning.

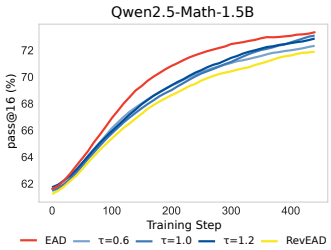


Figure 9: Reverse Annealing results. A reverse schedule fails to surpass standard temperature sampling.

Hyperparameter Sensitivity, Scheduling Effects and Stress Test. We ablate key hyperparameters of EAD in Appendix C, with two findings: (i) the *intra-sequence* annealing schedule is the primary driver—even with a fixed decay rate ($d_0 = d_{\max} = 25$), EAD still beats the best fixed-temperature baseline; and (ii) the *global-step-aware* decay rate adds complementary gains by keeping the schedule calibrated as responses lengthen, without which performance eventually drops. Overall, EAD is robust across a wide range of decay cap $d_{\max}$ and growth factor α values (Fig. 11a and 11b). We further stress-test EAD on the challenging AIME 2024 benchmark using the DAPO recipe with Qwen-3-8B-Base (Yang et al., 2025), confirming consistent improvements over fixed-temperature sampling (Appendix D).

Practical Guide: Not All Hyperparameters Are Equally Important. Although EAD exposes six hyperparameters ($\tau_{\max}, \tau_{\min}, c, d_0, \alpha, d_{\max}$), it does not enlarge the delicate part of RL tuning: as a *sampling-side* change it leaves the optimizer, loss, clipping, and learning rate untouched, so its settings do not interact with the brittle optimization hyperparameters. Five of the six are fixed defaults reused unchanged across every model, size, and RL algorithm (Table 2) and are robust in our ablations—$\tau_{\max}$=1.2, c=10, d_0=25, α=5, $d_{\max}$=4×10^4 (performance is flat for $d_{\max} \in [10^3, 10^5]$ and any positive α; Fig. 11a and 11b). The only model-dependent hyperparameter is the floor $\tau_{\min}$, set by a simple heuristic (0.1 for $\leq$1.5B, 0.6–0.8 for 7B+ models that already decode near-greedily). All reported gains use these untuned defaults, so EAD adds a single robust hyperparameter rather than expanding the tuning burden; see Appendix H for the full guide.

6 Related Work

Exploration in RLVR. Reinforcement learning with verifiable rewards (RLVR) (Shao et al., 2024; Guo et al., 2025; Lambert et al., 2024; Yu et al., 2025; Yang et al., 2025; Hu et al., 2025; Guan et al., 2025; Zeng et al., 2025) has become a major paradigm for improving LLM reasoning, with a growing literature on training algorithms (Yu et al., 2025; Liu et al., 2025; Yue et al., 2025b; Zheng et al., 2025a), reward design (Zuo et al., 2025; Zhao et al., 2025b; Agarwal et al., 2025; Prabhudesai et al., 2025), and evaluation methodology (Yue et al., 2025a; Zhao et al., 2025a; Hochlehnert et al., 2025). A central and open challenge is balancing exploration and exploitation along long reasoning trajectories (Cheng et al., 2025; Cui et al., 2025; Wang et al., 2025b; Deng et al., 2025). Recent work addresses this through diverse mechanisms: per-token entropy-guided sampling (Cheng et al., 2025), pass@k-targeted training objectives (Tang et al., 2025; Chen et al., 2025), high-covariance token control to prevent entropy collapse (Cui et al., 2025), selective updating of high-entropy minority

tokens (Wang et al., 2025b), and tree-based or selective rollout strategies (Li et al., 2025; Xu et al., 2025; Zheng et al., 2025b; Dou et al., 2025). However, these approaches often require tracking additional statistics or multi-threaded generation (Pan et al., 2025), adding computational overhead. In contrast, this paper presents the first systematic analysis of sampling temperature scheduling for RLVR and introduces a purely sampling-level annealed schedule that front-loads exploration and progressively stabilizes generation, requiring zero additional computation.

The temperature annealing design of EAD is reminiscent of *Simulated Annealing* (SA) (Kirkpatrick et al., 1983; Bertsimas & Tsitsiklis, 1993), which also uses a cooling temperature to transition from exploration to exploitation. Prior SA-inspired approaches in ML invariably apply a uniform temperature across all positions in a generated sequence (Liu et al., 2021; Zhang et al., 2024; Israel et al., 2025; Manvi et al., 2024). Our work departs from this convention by introducing an **intra-sequence annealed temperature** schedule, where the temperature varies dynamically within the generation of a single sequence. This novel approach allows for more nuanced control over exploration and leads to significant performance gains in RLVR. We provide a more detailed review of SA in Appendix E.

7 Discussion

We introduce Exploratory Annealed Decoding (EAD), a simple yet powerful sampling strategy for RLVR that employs a temperature-annealing schedule—exploring broadly early and converging to high-quality completions later. This schedule improves sampling diversity while controlling variance across rollouts, and applies at test time to enhance inference-time scaling. This simplicity is deliberate: unlike strategies requiring multi-threading (Pan et al., 2025) or tracking additional statistics (Cui et al., 2025), EAD adds zero computational overhead, integrates seamlessly with standard pipelines, and consistently outperforms baselines across diverse models and RL algorithms.

Limitations and Future Directions. Despite the encouraging results, several limitations suggest directions for future work: *Domain scope.* Our evaluation focuses on mathematical reasoning across diverse model families (LLaMA, Qwen) and sizes (1B–8B). While "explore early, exploit late" is grounded in a general property of autoregressive generation (§ 3), extending EAD to other verifiable domains (e.g., code) and larger scales remains important. We do not claim position-only early exploration is universally optimal: for code or agentic tasks, late tokens can be explicit decision points. EAD also does not eliminate late-stage exploration—because $\tau_{\min} > 0$, meaningful stochasticity persists throughout generation—but when critical decision points occur at unpredictable positions, a *prompt-* or *step-aware* schedule is a natural extension. *Assumption scope.* The "explore early" principle reflects an *expected* entropy trend; individual sequences may contain critical late-stage decisions (e.g., a final numerical answer). EAD mitigates this via a non-zero floor $\tau_{\min}$, but an adaptive schedule targeting high-uncertainty positions regardless of location could help further. *Evaluation metrics.* We primarily use multi-sample metrics (Pass@k, Worst@k, Majority@N), the natural choice for RLVR; the interplay with single-sample quality (Appendix J) and combining EAD with complementary methods (§ 6) are promising. *Hyperparameter sensitivity.* While EAD is robust across a wide range of hyperparameters (Appendix C), the optimal $\tau_{\min}$ varies with model capability (0.1 for 1B, 0.8 for 7B); a systematic study of how schedule parameters scale with model size would facilitate adoption (see the practitioner selection guide in Appendix H). *Schedule shape.* The negative-exponential form is one instance of position-aware annealed sampling, not necessarily optimal; we leave comparison of alternatives (e.g., linear, two-stage, or mean-matched controls that separate schedule *shape* from total *entropy budget*) to future work (Appendix I).

In summary, EAD provides a simple yet general way to couple exploration with the inherent progression of language generation. By reducing algorithmic overhead while improving trajectory quality, it opens new avenues for both efficient and effective RLVR.

Reproducibility Statement

To ensure the reproducibility of our results, we provide comprehensive details of our methodology and experimental setup. **Algorithm details** for Exploratory Annealed Decoding (EAD), including the annealing schedule and importance sampling correction, are described in § 4. Our **experimental setup**, including models, datasets, and training frameworks, is detailed in § 5.1. Specific **hyperparameters** for all experiments are provided in § 5.1 and further detailed in Appendix A. We detail

our **evaluation metrics** (e.g., Pass@16, Worst@16) and protocols throughout § 5. The relationship between temperature and entropy, central to our motivation, is formally proven in Appendix F. All experiments were conducted using publicly available models and datasets, with references provided in § 5.1. We plan to release our code in the formal version to facilitate further research.

USAGE OF LARGE LANGUAGE MODELS

In this work, besides running LLMs as experimental subjects, we used generative AI tools for: (i) editing the manuscript to improve readability (ChatGPT); (ii) literature search and related-work discovery (Gemini, ChatGPT); and (iii) assisting with code writing and debugging (Claude). We did not use generative AI tools to generate data, formulate claims, or write proofs. All AI-assisted text, citations, and code were reviewed and verified by the authors. We take responsibility for the final content of this work.

REFERENCES

David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985. ISSN 0364-0213. doi: https://doi.org/10.1016/S0364-0213(85)80012-4. URL https://www.sciencedirect.com/science/article/pii/S0364021385800124.

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*, 2025.

Edward Beeching, Shengyi Costa Huang, Albert Jiang, Jia Li, Benjamin Lipkin, Zihan Qina, Kashif Rasul, Ziju Shen, Roman Soletskyi, and Lewis Tunstall. Numinamath 7b cot. https://huggingface.co/AI-MO/NuminaMath-7B-CoT, 2024.

Glen Berseth. Is exploration or optimization the problem for deep reinforcement learning? *arXiv preprint arXiv:2508.01329*, 2025.

Dimitris Bertsimas and John Tsitsiklis. Simulated annealing. *Statistical science*, 8(1):10–15, 1993.

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

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

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

Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

Jia Deng, Jie Chen, Zhipeng Chen, Daixuan Cheng, Fei Bai, Beichen Zhang, Yinqian Min, Yanzipeng Gao, Wayne Xin Zhao, and Ji-Rong Wen. From trial-and-error to improvement: A systematic analysis of llm exploration mechanisms in rlvr. *arXiv preprint arXiv:2508.07534*, 2025.

Shihan Dou, Muling Wu, Jingwen Xu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. Improving rl exploration for llm reasoning through retrospective replay. *arXiv preprint arXiv:2504.14363*, 2025.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence. *arXiv preprint arXiv:2508.15260*, 2025.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*, 2025.

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.

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

James J Heckman, Hidehiko Ichimura, and Petra Todd. Matching as an econometric evaluation estimator. *The review of economic studies*, 65(2):261–294, 1998.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Jacob Hilton, Karl Cobbe, and John Schulman. Batch size-invariance for policy optimization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17086–17098. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/6ceb6c2150bbf46fd75528a6cd6be793-Paper-Conference.pdf.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *arXiv preprint arXiv:2504.07086*, 2025.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Zhenyu Hou, Xin Lv, Rui Lu, Jiajie Zhang, Yujiang Li, Zijun Yao, Juanzi Li, Jie Tang, and Yuxiao Dong. T1: Advancing language model reasoning through reinforcement learning and inference scaling. In *Forty-second International Conference on Machine Learning*, 2025.

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.

Daniel Israel, Aditya Grover, and Guy Van den Broeck. Enabling autoregressive models to fill in masked tokens. *arXiv preprint arXiv:2502.06901*, 2025.

Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

Pawel Ladosz, Lilian Weng, Minwoo Kim, and Hyondong Oh. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22, 2022.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Yizhi Li, Qingshui Gu, Zhoufutu Wen, Ziniu Li, Tianshun Xing, Shuyue Guo, Tianyu Zheng, Xin Zhou, Xingwei Qu, Wangchunshu Zhou, et al. Treepo: Bridging the gap of policy optimization and efficacy and inference efficiency with heuristic tree-based modeling. *arXiv preprint arXiv:2508.17445*, 2025.

Baohao Liao, Xinyi Chen, Sara Rajaee, Yuhui Xu, Christian Herold, Anders Søgaard, Maarten de Rijke, and Christof Monz. Lost at the beginning of reasoning. *arXiv preprint arXiv:2506.22058*, 2025.

Xianggen Liu, Pengyong Li, Fandong Meng, Hao Zhou, Huasong Zhong, Jie Zhou, Lili Mou, and Sen Song. Simulated annealing for optimization of graphs and sequences. *Neurocomputing*, 465: 310–324, 2021.

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.

Rohin Manvi, Anikait Singh, and Stefano Ermon. Adaptive inference-time compute: Llms can predict if they can do better, even mid-generation. *arXiv preprint arXiv:2410.02725*, 2024.

OpenAI. Learning to reason with llms. https://openai.com/index/learning-to-reason-with-llms/, 2024. Accessed: 2025-05-01.

Jiayi Pan, Xiuyu Li, Long Lian, Charlie Snell, Yifei Zhou, Adam Yala, Trevor Darrell, Kurt Keutzer, and Alane Suhr. Learning adaptive parallel reasoning with language models. *arXiv preprint arXiv:2504.15466*, 2025.

Ethan Perez, Sam Ringer, Kamile Lukosiute, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, et al. Discovering language model behaviors with model-written evaluations. In *Findings of the association for computational linguistics: ACL 2023*, pp. 13387–13434, 2023.

Mihir Prabhudesai, Lili Chen, Alex Ippoliti, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. Maximizing confidence alone improves reasoning. *arXiv preprint arXiv:2505.22660*, 2025.

Matthew Renze. The effect of sampling temperature on problem solving in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 7346–7356, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.432. URL https://aclanthology.org/2024.findings-emnlp.432/.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27 (3):379–423, 1948.

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.

Vaishnavi Shrivastava, Ahmed Awadallah, Vidhisha Balachandran, Shivam Garg, Harkirat Behl, and Dimitris Papailiopoulos. Sample more to think less: Group filtered policy optimization for concise reasoning. *arXiv preprint arXiv:2508.09726*, 2025.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Ava Spataru, Eric Hambro, Elena Voita, and Nicola Cancedda. Know when to stop: A study of semantic drift in text generation. *arXiv preprint arXiv:2404.05411*, 2024.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Yunhao Tang, Kunhao Zheng, Gabriel Synnaeve, and Remi Munos. Optimizing language models for inference time objectives using reinforcement learning. In *Forty-second International Conference on Machine Learning*, 2025.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Sebastian B Thrun. *Efficient exploration in reinforcement learning*. Carnegie Mellon University, 1992.

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

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. Octothinker: Mid-training incentivizes reinforcement learning scaling. *arXiv preprint arXiv:2506.20512*, 2025c.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Lilian Weng. Reward hacking in reinforcement learning. *lilianweng.github.io*, Nov 2024. URL https://lilianweng.github.io/posts/2024-11-28-reward-hacking/.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

Yixuan Even Xu, Yash Savani, Fei Fang, and Zico Kolter. Not all rollouts are useful: Down-sampling rollouts in llm reinforcement learning. *arXiv preprint arXiv:2504.13818*, 2025.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

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

Chenghao Yang, Sida Li, and Ari Holtzman. Llm probability concentration: How alignment shrinks the generative horizon. *Transactions on Machine Learning Research*, 2026.

Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient rl framework secretly brings you off-policy rl training, August 2025. URL https://fengyao.notion.site/off-policy-rl.

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

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025a.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025b.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

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

Rosie Zhao, Alexandru Meterez, Sham Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. Echo chamber: Rl post-training amplifies behaviors learned in pretraining. *arXiv preprint arXiv:2504.07912*, 2025a.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. *arXiv preprint arXiv:2505.19590*, 2025b.

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

Haizhong Zheng, Yang Zhou, Brian R Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. *arXiv preprint arXiv:2506.02177*, 2025b.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *AAAI*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, et al. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*, 2025.

A Minimal-RL Training Details

We mainly follow the Minimal-RL recipe (Xiong et al., 2025) in our experiments to ensure a fair comparison among different rollout sampling strategies. Specifically, we set a series of hyperparameters as in Table 1:

Hyperparameter	Value(s)
Training Batch Size	1024
Max Prompt Length	1024
Max Response Length	3072
Mini Batch Size	256
Micro Batch Size Per GPU	4
Learning Rate	10^{-6}

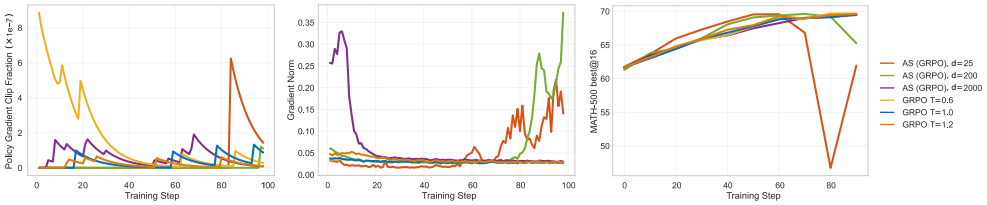
Table 1: Hyperparameter Setup for Running Minimal-RL recipe.

B Off-policy Issue and Truncated Importance Sampling Correction

B.1 Sampling Techniques Can Introduce Off-Policy Issue

One subtle yet troublesome drawback of reinforcement learning with sampling techniques is that it simultaneously introduces the *off-policy* problem: there is a gap between the behavior policy (used

for sampling) and the target policy (being optimized and parametrized by θ). This might introduce instability to the training and cause it to fail (See example training of RLVR with EAD Fig. 10).



(a) Clip Fraction Surge (b) Gradient Norm Surge. (c) Drastic Best@16 Drop

Figure 10: Off-policy samples bring training instability. The base model is Qwen2.5-Math-1.5B.

Noted that this off-policy phenomenon widely exists for any efficient sampling framework (Yao et al., 2025) and sampling strategy (for instance, when applying best-of-n sampling (Xiong et al., 2025) or filtering out responses (Shrivastava et al., 2025), the underlying distribution of responses is implicitly altered). To be more precise, we take the policy gradient loss as an example:

$$\mathbb{E}_{x\sim\mathcal{D},y\sim\pi^{\text{sampling}}_{\theta_{\text{old}}}(\cdot|x)}\left[\frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}A(y;x)\right]=\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_{\theta_{\text{old}}}(\cdot|x)}\left[\frac{\pi^{\text{sampling}}_{\theta_{\text{old}}}(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}\times\frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}A(y;x)\right],$$

where $\pi^{\text{sampling}}_{\theta_{\text{old}}}(\cdot\mid x)$ represents the underlying sampling distribution. In such case, an extra weight is implicitly added to each response in addition to its advantage $A(y;x)$.

This extra weight can significantly inflate the variance of the policy gradient, posing a stability challenge that our proposed EAD needs to mitigate. To quantify this effect in our proposed EAD, we now analyze such variance under a standard, fixed **temperature sampling**.

We use $\tau=1$ to define a policy π and consider the effect of τ on the variance of the gradient estimator. We reduce the problem to analyzing the variance inflation factor

$$\mathbb{E}_{y\sim\pi(\cdot|x;\tau)}\left[\frac{\pi(y\mid x;1)^2}{\pi(y\mid x;\tau)^2}\right]. \tag{3}$$

We begin with one-token case. Let o_i denote the ith token in the vocabulary V and h_i is its logit. Then equation 3 can be rewritten as

$$\sum_{i=1}^{|V|}\frac{h_i/(\sum_{j=1}^{|V|}h_j)}{h_i^{1/\tau}/(\sum_{j=1}^{|V|}h_j^{1/\tau})}\times\frac{h_i}{\sum_{j=1}^{|V|}h_j}=\frac{\sum_{i=1}^{|V|}h_i^{2-1/\tau}\sum_{i=1}^{|V|}h_i^{1/\tau}}{\left(\sum_{i=1}^{|V|}h_i\right)^2}$$

Proposition B.1. *Suppose $h_i\in[0,1]$ for all $i\in V$. $\sum_{i=1}^{|V|}h_i^{2-1/\tau}\sum_{i=1}^{|V|}h_i^{1/\tau}$ is decreasing when $\tau\le 1$ and increasing when $\tau\ge 1$, which implies it has a global minimum at $\tau=1$.*

Proof. Let $x=1/\tau$. We define

$$f(x)=\log\left(\sum_{i=1}^{|V|}h_i^{2-x}\right)+\log\left(\sum_{i=1}^{|V|}h_i^{x}\right).$$

Its derivative is

$$f'(x)=\frac{-\sum_{i=1}^{|V|}h_i^{2-x}\log h_i}{\sum_{i=1}^{|V|}h_i^{2-x}}+\frac{\sum_{i=1}^{|V|}h_i^{x}\log h_i}{\sum_{i=1}^{|V|}h_i^{x}}.$$

To analyze the sign of $f'(x)$, we define a helper function $g(x)=\frac{\sum_{i=1}^{|V|}h_i^{x}\log h_i}{\sum_{i=1}^{|V|}h_i^{x}}$. Then, $f'(x)=g(x)-g(2-x)$ and its sign depends on whether $g(x)$ is greater than, less than, or equal to $g(2-x)$. We take a look at derivative of g:

$$g'(x)=\frac{\left(\sum_{i=1}^{|V|}h_i^{x}(\log h_i)^2\right)\left(\sum_{i=1}^{|V|}h_i^{x}\right)-\left(\sum_{i=1}^{|V|}h_i^{x}\log h_i\right)^2}{\left(\sum_{i=1}^{|V|}h_i^{x}\right)^2}\ge 0.$$

Hence, g is an increasing function and

$$\begin{cases} f'(x) = g(x) - g(2-x) \geq 0, & \text{when } x \geq 1 \\ f'(x) = g(x) - g(2-x) = 0, & \text{when } x = 1 \\ f'(x) = g(x) - g(2-x) \leq 0, & \text{when } x \leq 1. \end{cases}$$

Accordingly, f is increasing when $x \geq 1$ and is decreasing when $x \leq 1$. Then the proposition easily follows. □

The same conclusion can be proved for multiple-token sequence by induction. Therefore, we get that when the temperature is far from 1, the off-policy issue could be severe and lead to large variance of the gradient estimator.

B.2 Truncated Importance Sampling Ratio Correction

To cancel such bias, an importance sampling ratio can be introduced (Hilton et al., 2022; Yao et al., 2025):

$$\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_{\theta_{\text{old}}}(\cdot|x)}\left[\frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}A(y;x)\right] = \mathbb{E}_{x\sim\mathcal{D},y\sim\pi^{\text{sampling}}_{\theta_{\text{old}}}(\cdot|x)}\left[\frac{\pi_{\theta_{\text{old}}}(y\mid x)}{\pi^{\text{sampling}}_{\theta_{\text{old}}}(y\mid x)}\times\frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}A(y;x)\right].$$

To further prevent negative effects by the extreme likelihood ratios and boost training stability, we truncate the likelihood ratio with an upper bound. That is, *truncated importance sampling* technique (Heckman et al., 1998). Taking the vanilla policy gradient loss as an example, the modified loss for EAD is as follows:

$$\mathbb{E}_{x\sim\mathcal{D},y\sim\pi^{\text{EAD}}_{\theta_{\text{old}}}(\cdot|x)}\left[\min\left(\frac{\pi_{\theta_{\text{old}}}(y\mid x)}{\pi^{\text{EAD}}_{\theta_{\text{old}}}(y\mid x)},\varepsilon\right)\frac{\pi_\theta(y\mid x)}{\pi_{\theta_{\text{old}}}(y\mid x)}A(y;x)\right].$$

Practical choices and bias–variance trade-off. We make the following choices explicit.

- **Sequence-level, not token-level.** The TIS ratio $\pi_{\theta_{\text{old}}}(y\mid x)/\pi^{\text{EAD}}_{\theta_{\text{old}}}(y\mid x)$ is computed once per sequence y on the same trajectories used for the policy update, rather than per token.
- **Truncation threshold.** We use $\varepsilon = 2.0$ throughout, following the empirical recommendation of Yao et al. (2025).
- **No normalization.** We apply the raw min-truncated ratio; the weights are not re-normalized within a batch.
- **Algorithm-agnostic integration.** TIS enters as a multiplicative weight on the policy-gradient loss. The ratio is evaluated under the behavior policy $\pi^{\text{EAD}}_{\theta_{\text{old}}}$ and therefore carries *no* gradient; only the inner policy ratio $\pi_\theta/\pi_{\theta_{\text{old}}}$ is differentiable. The same weight plugs into DAPO, GRPO, and EntropyMech without any per-algorithm tuning.

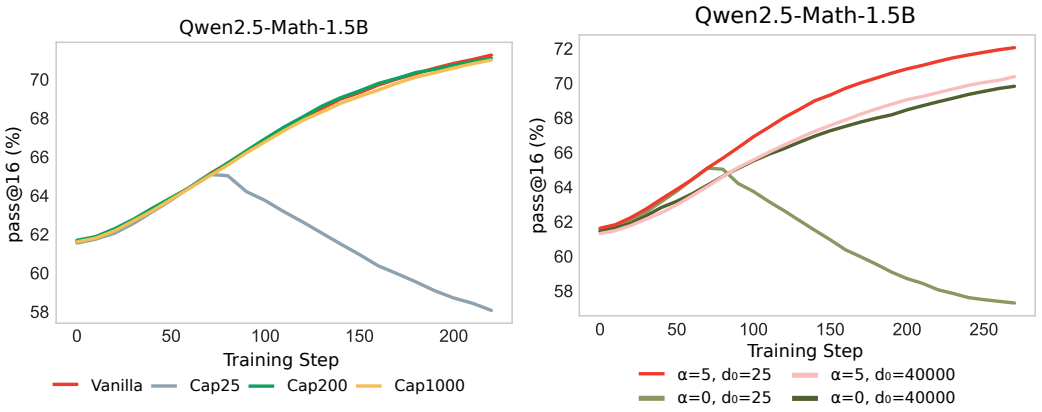
Importantly, truncation makes the estimator *biased*: the unbiased estimator is plain importance sampling (IS), and clipping at ε introduces a deterministic downward bias on the high-ratio tail in exchange for bounding the per-sample variance by $O(\varepsilon^2)$. Hence TIS should be read as a *variance-control mechanism* rather than an unbiased correction. The two limiting cases are instructive: as $\varepsilon \to \infty$, TIS recovers unbiased IS but reintroduces the destabilizing variance shown in Fig. 10; as $\varepsilon \to 1$, it collapses toward the naive uncorrected estimator. The chosen $\varepsilon = 2.0$ sits between these extremes. Empirically the truncation is rarely active—fewer than 1% of sequences hit the threshold in our runs—so TIS removes the rare, high-variance tail events that drive the clip-fraction surge and gradient-norm spike (Fig. 10a and 10b) while leaving the bulk of the gradient signal intact.

C Ablation Study: Hyperparameter Sensitivity and Scheduling Effects

Hyperparameter Sensitivity. We conduct an ablation study on the decay cap $d_{\max}$ and the growth factor α. As shown in Fig. 11a, EAD is robust to a wide range of $d_{\max}$ values, provided the cap is sufficiently large to ensure a smooth annealing schedule. Concretely, performance is stable for any $d_{\max} \geq 200$; the role of $d_{\max}$ is primarily to prevent failures caused by response-length growth during training. Conversely, a very small cap (e.g., $d_0 = d_{\max} = 25$) approximates a fixed schedule

(equivalent to $\alpha = 0$ throughout training). This setting causes an abrupt performance drop, likely because the model cannot adapt to the increasing response lengths during training (see Appendix G). In other words, the degenerate $d_{\max} = 25 = d_0$ case removes the intended global-step adaptation and effectively reduces EAD to a fixed-temperature run. Alternatively, using a large initial d_0 yields a smooth schedule even with $\alpha = 0$. However, while training remains stable, performance improves slowly due to overly conservative exploration, as shown in Fig. 11b. The present sweep covers the caps that are actually exercised within our training horizon; a wider multi-order-of-magnitude sweep ($d_{\max} \in \{200, 1000, 5000, 40000, 200000\}$) requires longer runs—within a short run the decay rate never reaches the larger caps—and is deferred for future work.

Intra-Sequence vs. Training-Step Scheduling. A natural question is whether the gains come from the *intra-sequence* annealing or merely from changing the effective temperature across training steps (via the global-step-aware decay rate). The ablation with $d_0 = d_{\max} = 25$ (Fig. 11a) isolates this: it retains the intra-sequence schedule but removes the training-step adaptation, and still outperforms the best fixed-temperature baseline, confirming that the within-sequence temperature schedule is the primary driver. The global-step-aware component provides further gains by keeping the schedule well-calibrated as response lengths grow.



(a) Decay cap $d_{\max}$.

(b) Growth factor α and initial decay d_0.

Figure 11: Ablation on hyperparameter sensitivity. **Left:** Ablation on decay cap $d_{\max}$. "Vanilla" denotes EAD with the default $d_{\max} = 40000$. Setting $d_{\max} = 25$ (where $d_0 = 25$) mimics a fixed temperature schedule (effectively $\alpha = 0$), leading to performance drops as the schedule fails to adapt to longer responses. EAD remains robust provided $d_{\max}$ allows for smooth temperature reduction (e.g., $d_{\max} \geq 200$). **Right:** Ablation on growth factor α and initial decay d_0. With a small d_0, a positive growth factor ($\alpha > 0$) is required to adapt to increasing response lengths and maintain a smooth intra-sequence schedule. While a large d_0 ensures smoothness even with $\alpha = 0$, it limits exploration, resulting in lower learning efficiency.

D Stress Test: DAPO-17K Recipe for AIME24

To verify the effectiveness of EAD on realistic mathematical reasoning tasks, we stress-test it using the DAPO recipe (Yu et al., 2025), training on DAPO-17k and evaluating on the challenging AIME 2024 benchmark. To accommodate GPU memory constraints, we utilize the Qwen-3-8B-Base (Yang et al., 2025) model (instead of 32B) and reduce the maximum response length from 20480 to 8192. We set $d_0 = 500$ and $d_{\max} = 40000$, and vary $\alpha \in \{5, 50\}$. As shown in Fig. 12, EAD achieves improved learning efficiency and consistently outperforms the standard temperature sampling baseline. We note that this experiment was conducted under reduced compute (8B model, shorter max length) and was not trained to full convergence; the results primarily demonstrate that EAD accelerates learning, with the final performance ceiling under full-scale training left for future work.

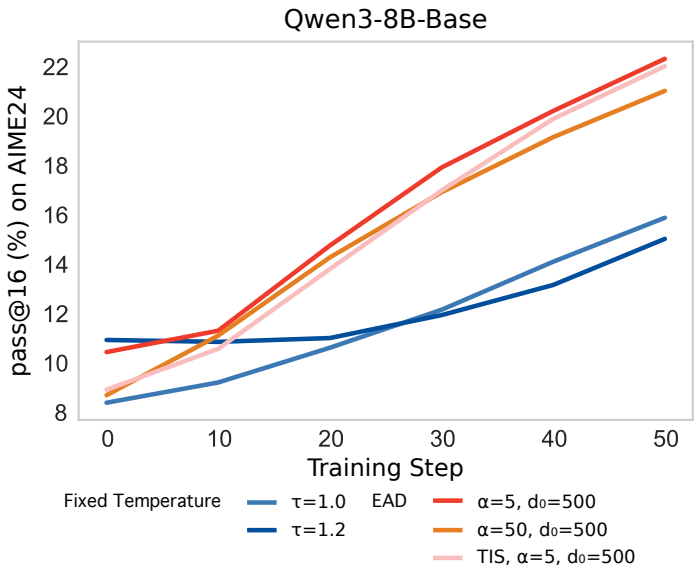


Figure 12: Stress-test of EAD using the DAPO recipe on AIME24. EAD outperforms the temperature sampling baseline across various hyperparameter setups.

E Connection to Simulated Annealing

Simulated Annealing (SA) is a probabilistic optimization technique inspired by annealing in metallurgy, designed to find the global optimum in a large search space (Kirkpatrick et al., 1983; Bertsimas & Tsitsiklis, 1993). The core principle involves a temperature parameter that controls the probability of accepting suboptimal states. Initially, a high temperature allows the search to escape local minima by exploring broadly (exploration). As the temperature gradually decreases, the algorithm increasingly favors better states, converging towards a high-quality solution (exploitation). This "coarse-to-fine" search dynamic, where high temperatures establish a solution's general structure and low temperatures refine its details, strongly parallels the generative process of LLMs (Yang et al., 2026). SA has been adapted in various machine learning contexts to manage the exploration-exploitation trade-off, including recent applications in graph optimization (Liu et al., 2021), text editing (Zhang et al., 2024), non-autoregressive generation (Israel et al., 2025), and efficient Best-of-N sampling (Manvi et al., 2024). However, these prior applications invariably apply a single, uniform temperature across all positions in a generated sequence. This approach fails to account for the heterogeneous roles of tokens at different positions (Wang et al., 2025b). Our work departs from this convention. To the best of our knowledge, we are the first to introduce an **intra-sequence annealed temperature** schedule, where the temperature varies dynamically within the generation of a single sequence. This novel approach allows for more nuanced control over exploration and leads to significant performance gains in RLVR.

F Proof of Temperature-Entropy Relationship

Proposition F.1. *The entropy of the softmax distribution is a non-decreasing function of the temperature $\tau > 0$.*

Proof. The strategy is to show that the entropy H is a non-increasing function of the inverse temperature $\beta = 1/\tau > 0$. The probability of sampling token v with temperature τ is given by the policy π_θ. For simplicity in the derivation, we denote this probability as $p_v(\tau)$:

$$p_v(\tau) \triangleq \pi_\theta(y_t = v \mid [x, y_{<t}]; \tau)$$

Let h_v be the logit for a token v in the vocabulary V. The probability of a token as a function of β is given by:

$$p_v(\beta) = \frac{\exp(\beta h_v)}{\sum_{v' \in V} \exp(\beta h_{v'})} \triangleq \frac{\exp(\beta h_v)}{Z(\beta)},$$

where $Z(\beta)$ is the partition function. The entropy, as a function of β, is:

$$H(\beta) = -\sum_{v \in V} p_v(\beta) \log p_v(\beta).$$

We can rewrite the entropy by substituting $\log p_v(\beta) = \beta h_v - \log Z(\beta)$:

$$\begin{aligned} H(\beta) &= -\sum_{v \in V} p_v(\beta)(\beta h_v - \log Z(\beta)) \\ &= \log Z(\beta) \left(\sum_{v \in V} p_v(\beta) \right) - \beta \sum_{v \in V} h_v p_v(\beta) \\ &= \log Z(\beta) - \beta \cdot \mathbb{E}_{v \sim p(\beta)}[h_v]. \end{aligned}$$

Now, we differentiate $H(\beta)$ with respect to β. Let $\bar{h}(\beta) = \mathbb{E}[h_v]$.

$$\frac{dH}{d\beta} = \frac{d}{d\beta}(\log Z(\beta)) - \frac{d}{d\beta}(\beta \bar{h}(\beta)).$$

First, we find the derivative of the log-partition function:

$$\frac{d}{d\beta}(\log Z(\beta)) = \frac{Z'(\beta)}{Z(\beta)} = \frac{\sum_v h_v \exp(\beta h_v)}{Z(\beta)} = \sum_v h_v p_v(\beta) = \bar{h}(\beta).$$

Next, we use the product rule for the second term:

$$\frac{d}{d\beta}(\beta \bar{h}(\beta)) = \bar{h}(\beta) + \beta \frac{d\bar{h}}{d\beta}.$$

Combining these gives:

$$\frac{dH}{d\beta} = \bar{h}(\beta) - \left(\bar{h}(\beta) + \beta \frac{d\bar{h}}{d\beta} \right) = -\beta \frac{d\bar{h}}{d\beta}.$$

The derivative $\frac{d\bar{h}}{d\beta}$ is the variance of the logits. We can show this by differentiating $\bar{h}(\beta)$:

$$\begin{aligned} \frac{d\bar{h}}{d\beta} &= \frac{d}{d\beta} \left(\frac{\sum_v h_v \exp(\beta h_v)}{Z(\beta)} \right) \\ &= \frac{(\sum_v h_v^2 \exp(\beta h_v)) Z(\beta) - (\sum_v h_v \exp(\beta h_v)) Z'(\beta)}{Z(\beta)^2} \\ &= \sum_v h_v^2 p_v(\beta) - \left(\sum_v h_v p_v(\beta) \right) \left(\frac{Z'(\beta)}{Z(\beta)} \right) \\ &= \mathbb{E}[h^2] - (\mathbb{E}[h])^2 = \mathrm{Var}_{v \sim p(\beta)}(h_v). \end{aligned}$$

Substituting this back, we arrive at the final expression for the derivative of entropy:

$$\frac{dH}{d\beta} = -\beta \cdot \mathrm{Var}_{v\sim p(\beta)}(h_v).$$

By definition, the temperature $\tau > 0$, so the inverse temperature $\beta > 0$. The variance of any random variable is non-negative. This can be formally shown using **Jensen's inequality**: for the convex function $\phi(x) = x^2$, we have $\mathbb{E}[\phi(h)] \geq \phi(\mathbb{E}[h])$, which means $\mathbb{E}[h^2] \geq (\mathbb{E}[h])^2$, and thus $\mathrm{Var}(h) \geq 0$.

Therefore, the derivative of entropy with respect to β is non-positive:

$$\frac{dH}{d\beta} = \underbrace{-\beta}_{\leq 0} \cdot \underbrace{\mathrm{Var}(h_v)}_{\geq 0} \leq 0.$$

Since $H(\beta)$ is a non-increasing function of β, and β is inversely proportional to T, it follows that $H(\tau)$ must be a non-decreasing function of the temperature τ. □

G Increasing Length in RL Training

During RL training, our algorithm (EAD) incentivizes the model to generate longer, more effective reasoning chains for difficult problems, especially for 7B models (Fig. 13). While both EAD and temperature sampling initially learn to shorten their responses by shifting from complex code-based solutions to direct mathematical reasoning, their behavior later diverges. The response length from temperature sampling stabilizes, whereas EAD learns to selectively increase reasoning length for harder problems, which boosts final performance. For these experiments, EAD is applied in the DAPO algorithm for sampling rollouts. We use the same training setup as detailed in § 5.

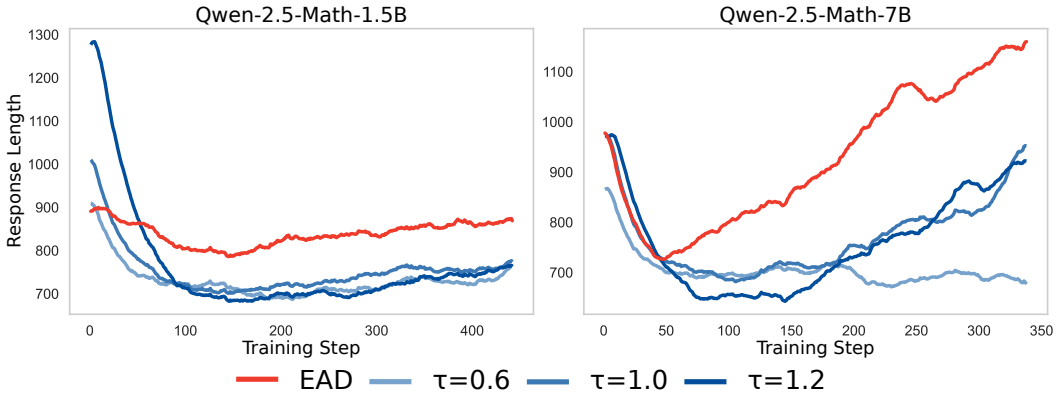


Figure 13: Compared with normal temperature sampling, EAD can naturally incentivize the model to generate longer reasoning chains.

H Hyperparameter Summary and Selection Guide

EAD exposes six hyperparameters: the maximum and floor temperatures $(\tau_{\max}, \tau_{\min})$, the template cutoff c, and the global-step-aware decay parameters $(d_0, \alpha, d_{\max})$. Table 2 lists the exact values used in every experiment; the only model-dependent hyperparameter is $\tau_{\min}$.

Selection guide for practitioners. Most of these values transfer directly; in a new setting the only hyperparameter that typically needs tuning is $\tau_{\min}$.

- $\tau_{\max}$: the highest value that does not hurt single-sample quality on a small held-out probe; 1.2 worked for every model we tried.
- $\tau_{\min}$: scales with the sharpness of the base model's next-token distribution; 0.1 is safe for 1B–1.5B models, while 0.5–0.8 suits stronger 7B+ models that already decode near-greedily. A principled choice is to inspect the entropy of the *base* model on the late tokens of a small probe set.

Setting / Model	$\tau_{\max}$	$\tau_{\min}$	c	d_0	α	$d_{\max}$
Qwen-2.5-Math-1.5B (main)	1.2	0.1	10	25	5	40000
Llama-3.2-1B-Instruct (main)	1.2	0.1	10	25	5	40000
Qwen-2.5-Math-7B	1.2	0.8	10	25	5	40000
Inference-time (off-the-shelf Qwen-2.5)	1.2	0.1	10	25†	–	–

Table 2: Default EAD hyperparameters used across experiments. †At inference time there is no training step, so we use a fixed decay rate $d = 25$ (no global-step adaptation, hence α and $d_{\max}$ do not apply). The stress test in Appendix D instead uses $d_0 = 500$, $d_{\max} = 40000$, and $\alpha \in \{5, 50\}$.

- c: a small constant (10–20 tokens) covering chat-formatting / template prefixes.
- d_0: set so that the first $\sim$100 tokens remain at $\tau_t \geq 1.0$.
- α: any positive growth factor; $\alpha = 5$ (one extra "hot" token per training step) sufficed for all runs.
- $d_{\max}$: robust across $[10^3, 10^5]$ (Fig. 11a); we use 4×10^4.

We leave a fully automatic, model-side prescription for $\tau_{\min}$ as a clean follow-up.

I Schedule Shape: Scope and Alternative Schedules

We deliberately frame EAD as a practical instance of *position-aware annealed rollout sampling*, not as evidence that the negative-exponential schedule is uniquely optimal. The broader idea of annealing—and variationally motivated views of adapting temperature or entropy during sampling and optimization (Kirkpatrick et al., 1983; Bertsimas & Tsitsiklis, 1993; Ziebart et al., 2008; Haarnoja et al., 2018)—is well established; our contribution is the demonstration that a simple position-aware, training-step-aware schedule is easy to plug into LLM RL rollouts and improves the exploration–quality trade-off in math RLVR.

We chose the negative-exponential form because it decays slowly at the very start of generation (where exploration is most valuable) and then cools smoothly toward a stable floor. Other monotonically decreasing schedules are natural candidates and may match EAD with appropriately tuned hyperparameters, including (i) a *linear* schedule $\tau_t = \max\{\tau_{\max} - \kappa t, \tau_{\min}\}$, (ii) a *two-stage* "hot-then-cold" schedule that holds $\tau_{\max}$ for the first block of tokens before dropping to $\tau_{\min}$, and (iii) a *mean-matched* control that fixes the average temperature over the sequence to isolate *schedule shape* from *entropy budget*. A systematic comparison of these alternatives—disentangling the effect of the schedule's shape from the total exploration budget it spends—is an interesting and valuable direction that we leave to future work; identifying the optimal scheduler is not the focus of this paper.

J Final-Checkpoint Results: Pass@1 and Reporting Protocol

Our primary metrics are multi-sample (Pass@k, Worst@k, Majority@N) because EAD targets *training-time* exploration, where the learner benefits from a diverse *set* of rollouts under a fixed budget K. A natural question is whether EAD also improves the single-sample policy, or only the diversity of the rollout set. To separate these, Table 3 reports the final-checkpoint Pass@1 (the cleanest cost-one metric) alongside the multi-sample Pass@16 and Worst@16 on the MATH500 evaluation set (500 problems; 16 samples per prompt for the @16 metrics). EAD improves Pass@1 over every fixed-temperature DAPO and GRPO baseline on *both* models, indicating that the gains are not solely a multi-sample diversity effect. On Qwen-2.5-Math-1.5B EAD is also strongest on Pass@16 and Worst@16; on the noisier Llama-3.2-1B-Instruct (note the wide eval-set bootstrap intervals on the @16 metrics) EAD is statistically on par with the strongest fixed-temperature baseline on Pass@16/Worst@16 while clearly exceeding all GRPO baselines.

Method	Pass@1	Pass@16	Worst@16
Qwen-2.5-Math-1.5B			
EAD (ours)	**72.53**	**74.15** $\pm$0.01	**73.99** $\pm$0.01
DAPO-T0.6	70.83	73.05 $\pm$0.00	73.00 $\pm$0.04
DAPO-T1.0	69.67	73.81 $\pm$0.00	73.70 $\pm$0.03
DAPO-T1.2	69.73	73.91 $\pm$0.01	73.79 $\pm$0.02
GRPO-T0.6	69.58	72.65 $\pm$0.00	72.63 $\pm$0.00
GRPO-T1.0	69.76	73.17 $\pm$0.01	73.05 $\pm$0.01
GRPO-T1.2	70.16	73.49 $\pm$0.01	73.27 $\pm$0.02
Llama-3.2-1B-Instruct			
EAD (ours)	**40.73**	46.76 $\pm$3.78	31.65 $\pm$1.89
DAPO-T0.6	39.80	45.42 $\pm$4.08	29.41 $\pm$2.37
DAPO-T1.0	38.66	49.48 $\pm$4.42	31.80 $\pm$2.51
DAPO-T1.2	38.38	48.39 $\pm$4.33	30.52 $\pm$2.66
GRPO-T0.6	36.38	39.64 $\pm$2.17	30.52 $\pm$1.31
GRPO-T1.0	36.72	39.81 $\pm$2.91	29.02 $\pm$1.60
GRPO-T1.2	36.38	42.06 $\pm$3.72	27.97 $\pm$2.15

Table 3: Final-checkpoint Pass@1, Pass@16, and Worst@16 (%) on the MATH500 evaluation set (500 problems). The @16 metrics use 16 samples per prompt and are reported with eval-set bootstrap 95% confidence intervals following the protocol of Hochlehnert et al. (2025); Pass@1 is a separate cost-one decode whose bootstrap standard deviation is below 0.01. EAD uses the default configuration of Table 2; bold marks the best entry in each column where EAD attains it. On Qwen EAD is best on all three metrics. On Llama the @16 intervals are wide ($\sim$4 points): EAD gives the best Pass@1 and overlaps the strongest fixed-temperature baseline on Pass@16/Worst@16, while substantially exceeding every GRPO baseline. The @16 values are the final points of the training curves in Fig. 3; a companion comparison of mean response length is in Appendix G (Fig. 13).