Efficient KL Divergence Estimation via Truncated Top-K Integration for Large Language Models

Xinyuan Wang* **Luozhijie Jin*** **Bo Wang** **Yuan Li**
Zhangyue Yin† **Xipeng Qiu†**
Fudan University
{xinyuanwang25,lzjjin22,bwang22,liyuan24,yinzy21}@m.fudan.edu.cn
xpqiu@fudan.edu.cn

Abstract

Kullback-Leibler (KL) divergence regularization is essential for stabilizing reinforcement learning from human feedback (RLHF) in large language models (LLMs), yet its exact computation requires summing over vocabularies of all tokens, incurring prohibitive memory costs during training. Existing stochastic estimators circumvent this bottleneck by estimating KL divergence using only the sampled token from the trajectory, but suffer from high variance (k_1) or systematic bias (k_2). We propose **TIKE** (Top-k Importance-weighted KL Estimator), which exploits the Zipfian structure of language model distributions: by deterministically integrating over only the top-k tokens, TIKE captures most of the probability mass while effectively reducing memory cost. To ensure correctness in off-policy settings characteristic of Group Relative Policy Optimization (GRPO), we incorporate importance sampling weights that correct for distribution shift between rollout and optimization policies. Experiments on models across diverse benchmarks demonstrate that TIKE consistently outperforms stochastic baselines, while exhibiting substantially lower gradient variance. Our analysis reveals that TIKE closely tracks the exact Rao-Blackwellized estimator with near-zero variance, offering a practical path toward stable, memory-efficient KL regularization for reasoning-intensive LLMs training. Code: https://github.com/jinluo12345/TIKE

1 Introduction

Reinforcement Learning with Verifiable Rewards (RLVR) has become the standard paradigm for enhancing the complex reasoning capabilities of Large Language Models (LLMs) (Deng et al., 2025; Liu et al., 2025b; Wen et al., 2025). A variety of optimization algorithms have been proposed, ranging from Proximal Policy Optimization (PPO) (Schulman et al., 2017), Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) to Group Relative Policy Optimization (GRPO) (Shao et al., 2024). However, a fundamental challenge in these reasoning tasks is the scarcity of supervision signals: models typically receive a reward only upon reaching the final outcome, lacking dense feedback for intermediate reasoning steps (Lyu et al., 2025; Cui et al., 2025; Zheng et al., 2025). Consequently, the stability of these algorithms relies heavily on the regularization term $D_{KL}(\pi_\theta||\pi_{\text{ref}})$ (Liu et al., 2025c), which serves as a crucial anchor to constrain the policy from deviating excessively from the reference distribution while exploring the sparse reward space (Xiong et al., 2023; Di Sipio, 2025).

The fundamental challenge in computing this regularization term stems from **prohibitive memory overhead**. A theoretically rigorous calculation of the KL divergence mandates integrating over the full vocabulary $\mathcal{V}$ at every timestep (Schulman, 2020). This requires retaining the complete probability distribution tensors and their associated gradients, causing memory consumption to scale linearly with the vocabulary size $|\mathcal{V}|$. For modern LLMs with extensive vocabularies ($|\mathcal{V}| > 100$k) operating on long reasoning contexts (Zeng et al., 2025; Dubey et al., 2024; Team et al., 2024, 2025), this dense computation becomes intractable on hardware. Consequently, prevalent methods compromise by approximating the divergence using only the single sampled token o_t (Ouyang et al., 2022; Stiennon et al., 2020).

Our approach is motivated by two key insights. First, we leverage the Zipfian nature of language model distributions (Tullo and Hurford, 2003; Zhemchuzhina et al., 2022): empirical analysis reveals that the cumulative probability mass is heavily concentrated in a small set of top-k tokens, rendering the contribution of the long tail negligible. Second, we strictly account for the off-policy

*Equal contribution.
†Corresponding authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8973–8985

nature of the training loop (Hu, 2025; Yu et al., 2025; Liu et al., 2025d). To correct for the distribution shift between the behavior policy π_{old} and the target policy π_θ, we employ importance sampling weights—a formulation aligned with recent findings by Deepseek Team (Liu et al., 2025a).

Building upon these, we propose **TIKE** (Top-k Importance-weighted KL Estimator), a hybrid estimator specifically designed for the reinforcement learning framework that achieves the "best of both worlds": it is (1) **Memory-Efficient** via Top-k truncation, avoiding the full vocabulary tensor; (2) **Low-Variance** via deterministic summation of the head of the distribution; and (3) **Unbiased** via importance sampling correction.

2 Preliminary

2.1 Exact KL Divergence in LLMs

In Reinforcement Learning from Human Feedback (RLHF), the Kullback-Leibler (KL) divergence (Van Erven and Harremos, 2014) serves as a critical regularization term. Let q denote the input query, o_t the token generated at step t, and $o_{<t}$ the preceding history. Without this constraint, the policy π_θ tends to diverge excessively from the reference π_{ref}, leading to reward hacking or catastrophic model collapse (Zhang et al., 2025).

The **exact** KL divergence at step t is defined as the expected log-likelihood ratio summed over the entire vocabulary $\mathcal{V}$:

$$D_{\text{KL}}(\pi_\theta(\cdot \mid q, o_{<t}) \| \pi_{\text{ref}}(\cdot \mid q, o_{<t})) = \sum_{v \in \mathcal{V}} \pi_\theta(v \mid q, o_{<t}) \log \frac{\pi_\theta(v \mid q, o_{<t})}{\pi_{\text{ref}}(v \mid q, o_{<t})} \quad (1)$$

Computing Eq. 1 requires iterating over the full vocabulary, incurring prohibitive memory costs.

2.2 Schulman's Estimators

To address this bottleneck, Schulman (2020) proposed estimators that approximate the KL using only the sampled token o_t rather than the full summation.

We define the probability ratio for the sampled token as $r = \frac{\pi_{\text{ref}}(o_t|q,o_{<t})}{\pi_\theta(o_t|q,o_{<t})}$. The estimators are:

- k_1 **(Standard log-ratio):** Defined as $-\log r$ (Schulman, 2020). While this estimator provides an unbiased estimate of the KL divergence, it suffers from extremely high variance, which can lead to noisy gradients and unstable training updates.
- k_2 **(Variance-reduced):** Defined as $k_2 = \frac{1}{2}(\log r)^2$ (Schulman, 2020). This estimator attempts to reduce variance via a second-order Taylor approximation. However, it is fundamentally biased and only serves as a valid approximation when π_θ is in close proximity to π_{ref}. Its reliability degrades during the active exploration phase of RL.
- k_3 **(Unbiased non-negative):** A general approach to reduce estimator variance is through control variates (Ross, 2002). Formally, a control variate is any function $g : \mathcal{V} \to \mathbb{R}$ for which $\mathbb{E}[g(x)]$ can be computed (Schulman, 2020). The control variate Monte Carlo estimator is defined as

$$\mu_{CV} = \frac{1}{M} \sum_{m=1}^{M} f(x) + \alpha \cdot (g(x) - \mathbb{E}[g(x)]) \quad (2)$$

where $\alpha \in \mathbb{R}$ is a calibration parameter and $f(x) \stackrel{\text{def}}{=} -\log r$. By defining the control variate as the likelihood ratio $g(x) = r$ and fixing the calibration parameter to $\alpha = 1$, the estimator simplifies to ($\mathbb{E}[r] = 1$):

$$k_3 = r - 1 - \log r \quad (3)$$

k_3 is the basic Monte Carlo KL estimator (k_1) plus a zero-mean control-variate term added to reduce variance without changing the expected value. The choice of $\alpha = 1$ is strategic rather than strictly variance-minimizing. It ensures the estimator possesses the geometric property of non-negativity, thereby preventing the numerical instability associated with negative KL estimates often observed in k_1.

Despite their widespread adoption, some experiments demonstrated that when used directly in the loss function (as is typical in GRPO), the k_3 estimator yields biased gradients that do not optimize the true KL objective, potentially leading to policy collapse (Shah et al., 2025). Meanwhile, k_1 remains too noisy for stable convergence in complex reasoning tasks.

3 Related Work

3.1 Exact Integration vs. Sampling

In contrast to sampling methods, some researchers proposed Rao-Blackwellized (RB) estimator (Amini et al., 2025), which computes the

exact expectation following Eq. 1. While theoretically optimal, this approach re-introduces the prohibitive computational and memory costs that sampled estimators sought to avoid. For models with large vocabularies, full Rao-Blackwellization is often impractical for long-context training.

3.2 Importance Sampling in KL Estimation

Most recently, the DeepSeek-V3.2 (Liu et al., 2025a) technical report identified another critical source of error in RL training: the *off-policy distribution shift*. In standard training loops, trajectories are sampled from an old policy π_{old} (inference engine) and reused for multiple optimization steps on the current policy π_θ. Since standard estimators implicitly assume on-policy sampling, Deepseek team proposed correcting this discrepancy by applying an importance sampling (IS) weight $r_t^{IS} = \pi_\theta(o_t \mid q, o_{<t})/\pi_{old}(o_t \mid q, o_{<t})$ to the estimator, ensuring the gradient is unbiased with respect to the current policy.

4 Methodology

When applying policy-gradient RL to large language models, unconstrained optimization can push the policy into regions far from the pretrained/SFT distribution, degrading fluency and causing unstable behavior. To stabilize learning, we regularize the RL objective with a KL-divergence term to a fixed reference policy, effectively acting as an "elastic tether" that discourages excessive drift while still allowing reward improvement. This KL-regularized formulation is standard in RLHF-style fine-tuning of language models and has been used to prevent the fine-tuned model from drifting too far from the pretrained baseline.

Figure 1 illustrates the core training dynamics of Group Relative Policy Optimization (GRPO). The main plot depicts the "tug-of-war" in parameter space: the model (π_θ) attempts to move toward the High Reward Peak, but is restrained by the KL Divergence "Rubber Band" anchored to the Reference Policy (π_{ref}) to prevent model collapse. The inset highlights the critical advantage of the Top-k algorithm. By truncating the long tail of the token distribution (grey zone), Top-k actively filters out low-probability, high-variance samples ("Tail Hits"). This ensures that the KL penalty is estimated solely from the reliable "Safe Zone" (green), drastically reducing estimator noise and preventing erratic updates caused by rare, irrelevant tokens.

4.1 The Zipfian Hypothesis and Truncation

Let $\mathcal{S}_t^k$ be the set of indices corresponding to the Top-k logits of the current policy $\pi_\theta(\cdot|q, o_{<t})$. Mathematically, if we sort the vocabulary by probability, the cumulative mass of the Top-k tokens ($k \ll \mathcal{V}$) approximates 1:

$$\sum_{v_t^i \in \mathcal{S}_t^k} \pi_\theta(v_t^i|q, o_{<t}) \approx 1 - \delta_t^k \tag{4}$$

where δ_t^k is negligible. This observation allows us to truncate the summation to a small set $\mathcal{S}_k$ (e.g., $k = 16$) without significant loss of precision. This reduces the memory complexity of the KL calculation from $\mathcal{O}(|\mathcal{V}|)$ to $\mathcal{O}(k)$, effectively solving the RAM bottleneck while retaining the stability benefits of exact integration.

Specifically, for the Qwen2.5 series (Team et al., 2024) with a vocabulary size $\mathcal{V} = 151,643$, a typical reinforcement learning setup often involves a batch size of 32 per GPU with a max-sequence length of 2,048. Such configurations result in a probability tensor of approximately 9.93×10^9 elements. In a full-vocabulary KL calculation, this requires 37.01 GB of GPU memory for FP32 or 18.51 GB for BF16/FP16 tensors, which often exceeds the available remaining memory during RL training. By contrast, our Top-16 truncation reduces the tensor size to only $32 \times 2048 \times 16$ elements, requiring merely 4.19 MB (FP32) or 2.10 MB (FP16), representing a memory reduction of over 99.9%.

To empirically validate the sparsity of the token distribution, we visualized the probability dynamics of generated sequences using prompts from the AIME24 dataset (Mathematical Association of America, 2025). Specifically, we extracted the probability values for each token step during inference. As illustrated in Figure 2, the extracted probabilities exhibit a clear pattern: while the probability of the single best token (green) can drop significantly at branching points requiring reasoning, the cumulative mass of the Top-16 tokens (red) is robustly stable near 1. This empirical observation supports the Zipfian hypothesis (Tullo and Hurford, 2003): natural language probability distributions, $\pi_\theta(\cdot|q, o_{<t})$, follow a heavy-tailed distribution where the mass is highly concentrated in a small number of valid next tokens.

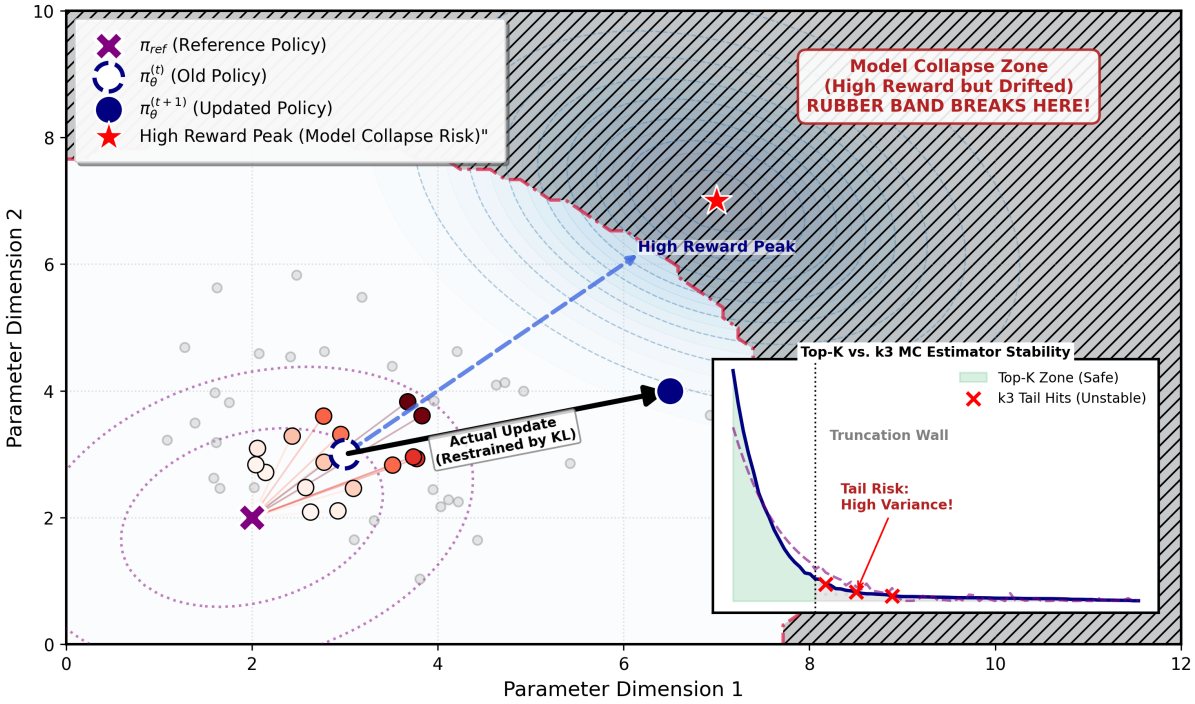

Figure 1: Illustration of the tension in GRPO between reward maximization and KL regularization. The policy update is pulled toward a high-reward but collapsed region (hatched), while the KL penalty acts as a "rubber band" restraining the update within a safe trust region. The inset compares Top-K and k_3 MC KL estimator stability, showing that the MC estimator suffers from high-variance tail hits beyond the truncation boundary.

4.2 Top-k Importance-weighted KL Estimator

In this section, we propose our **TIKE** (Top-k Importance-weighted KL Estimator). Crucially, simply minimizing this term would be incorrect because the history h was generated by the rollout policy π_{old}, not the current policy π_θ. To strictly correct for this off-policy distribution shift, we apply the importance sampling ratio r_t^{IS} (Liu et al., 2025a):

$$r_t^{IS} = \frac{\pi_\theta(o_t|q, o_{<t})}{\pi_{old}(o_t|q, o_{<t})} \tag{5}$$

We calculate the **deterministic partial KL** divergence over this truncated set:

$$D_t^{\text{TopK}}(\pi_\theta(o_t)\|\pi_{ref}(o_t)) = r_t^{IS} \sum_{v_t^i \in \mathcal{S}_t^k} \pi_\theta(v_t^i|q, o_{<t}) \log\left(\frac{\pi_\theta(v_t^i|q, o_{<t})}{\pi_{\text{ref}}(v_t^i|q, o_{<t})}\right) \tag{6}$$

To incorporate the TIKE estimator into the Group Relative Policy Optimization (GRPO) framework, we sample a group of G outputs $\{o_1, o_2, \ldots, o_G\}$ for each query q from the old policy π_{old}. Let R_i denote the reward obtained for the i-th output. We compute the advantages $\hat{A}_{i,t}$ by normalizing the rewards within each group:

$$\hat{A}_{i,t} = \frac{R_i - \mu\left(\{R_i\}_{i=1}^G\right)}{\sigma\left(\{R_i\}_{i=1}^G\right)} \tag{7}$$

The objective is clipped using a parameter ϵ to constrain policy updates, ensuring monotonicity:

$$A_{i,t}^{min} = \min\left(r_{i,t}(\theta)\,\hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\,\hat{A}_{i,t}\right) \tag{8}$$

Finally, we compute the total loss by averaging over the dataset distribution $q \sim P(Q)$ and the sequence length $|o_i|$ of each generated response where coefficient β controls the strength of the per-token KL penalty:

$$\mathcal{J}_{\text{TIKE}}^\theta = \mathbb{E}_{q\sim P(Q),\{o_i\}_{i=1}^G\sim\pi_{\text{old}}(\cdot|q)}\left[\frac{1}{G}\sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|}\left(A_{i,t}^{min} - \beta\, D_t^{\text{TopK}}\left(\pi_\theta(o_{i,t})\|\pi_{\text{ref}}(o_{i,t})\right)\right)\right] \tag{9}$$

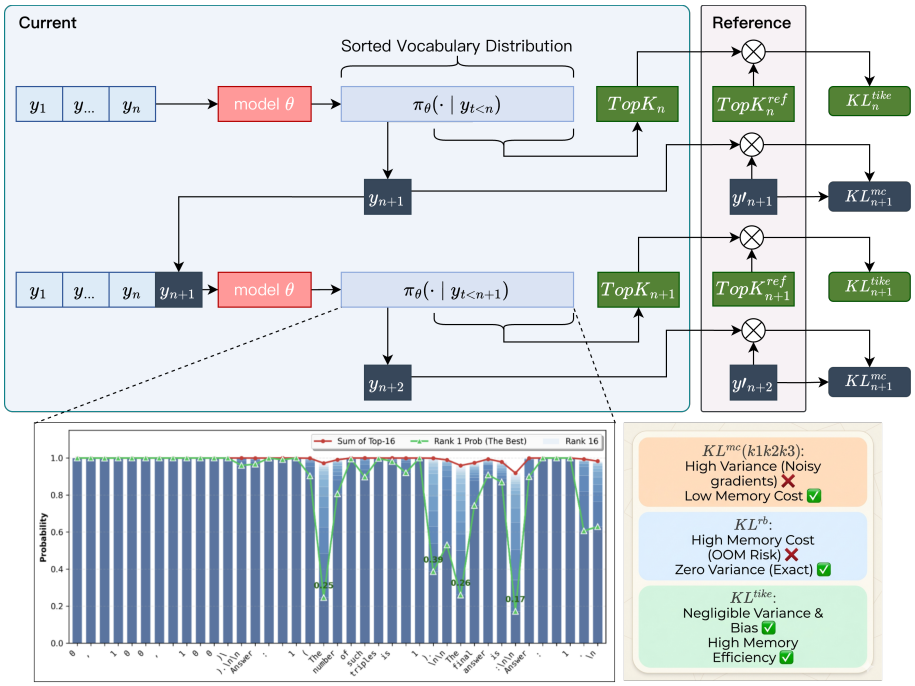

Figure 2: Overview of the TIKE framework. **(Top)** At each token position, TIKE extracts the top-k vocabulary mass from both the current and reference policies and combines them via importance weighting to compute a low-variance KL estimate KL^{tike}. **(Bottom-left)** Token-level probability profile showing that the top-16 mass (red) nearly saturates the full distribution at most positions, with uncertainty concentrated on a small fraction of semantically critical tokens (green). **(Bottom-right)** Qualitative comparison: KL^{mc} is memory-efficient but high-variance; KL^{rb} is exact but prohibitively memory-intensive; KL^{tike} achieves negligible variance and bias with high memory efficiency.

4.3 Theoretical Analysis

Let $D_t^{KL}(h_t) = D_t^{KL}(\pi_\theta(\cdot|h_t)\|\pi_{\text{ref}}(\cdot|h_t))$ be the true KL divergence at history $h_t \overset{\text{def}}{=} (q, o_{<t})$. Standard KL estimators (e.g., k_1, k_3) are computed from the sampled token $o_t \sim \pi_{\text{old}}(\cdot \mid h_t)$ and therefore require an importance weight to correct the *action-sampling* mismatch. In TIKE, since we deterministically integrate over a truncated action set, the remaining off-policy discrepancy primarily comes from the *state/history distribution* shift $d^{\pi_{\text{old}}}(h_t)$ vs. $d^{\pi_\theta}(h_t)$; correcting this exactly would require trajectory-/prefix-level importance weights (products over time), which are typically avoided for variance reasons in PPO/GRPO-style updates.

Top-k truncation and tail residual. Define the truncated KL (Top-k components) at h_t:

$$D_{\text{KL}}^k(h_t) \triangleq \sum_{v \in \mathcal{S}_t^k} \pi_\theta(v \mid h_t) \log \left(\frac{\pi_\theta(v \mid h_t)}{\pi_{\text{ref}}(v \mid h_t)} \right), \tag{10}$$

and the tail residual

$$\xi_{\text{tail}}(h_t) \triangleq \sum_{v \notin \mathcal{S}_t^k} \pi_\theta(v \mid h_t) \log \frac{\pi_\theta(v \mid h_t)}{\pi_{\text{ref}}(v \mid h_t)}. \tag{11}$$

Then the identity

$$D_{\text{KL}}(h_t) = D_{\text{KL}}^k(h_t) + \xi_{\text{tail}}(h_t) \tag{12}$$

holds for every h_t. Under the Zipfian hypothesis, the tail mass $\delta_t^k = \sum_{v \notin \mathcal{S}_t^k} \pi_\theta(v \mid h_t)$ is small so $\xi_{\text{tail}}(h_t)$ is typically negligible in practice, but it does not have a fixed sign in general.

Expectation of TIKE (properly conditioned). Recall our TIKE penalty (as used in the loss) is

$$D_t^{\text{TopK}} = r_t^{IS} \sum_{v_t^i \in \mathcal{S}_t^k} \pi_\theta(v_t^i \mid h_t) \log \left(\frac{\pi_\theta(v_t^i \mid h_t)}{\pi_{\text{ref}}(v_t^i \mid h_t)} \right) \tag{13}$$

For a *fixed* history h_t, the summation term is deterministic given $\pi_\theta(\cdot \mid h_t)$, hence

$$\mathbb{E}_{o_t \sim \pi_{\text{old}}(\cdot|h_t)}\left[D_t^{\text{TopK}} \mid h_t\right] = \left(\mathbb{E}_{o_t \sim \pi_{\text{old}}(\cdot|h_t)}[r_t^{IS} \mid h_t]\right) \cdot D_{\text{KL}}^k(h_t) \tag{14}$$

since $\mathbb{E}_{o_t\sim\pi_{\text{old}}}[r_t^{IS} \mid h_t] = 1$ whenever $\pi_{\text{old}}(\cdot \mid h_t)$ has support covering $\pi_\theta(\cdot \mid h_t)$. Therefore, TIKE exactly integrates the Top-k components *at a given history*. The remaining off-policy discrepancy is due to sampling histories from $d^{\pi_{\text{old}}}$ instead of d^{π_θ}, which is typically controlled in GRPO/PPO by frequent policy refresh and clipping, keeping π_θ close to π_{old} (Shao et al., 2024).

Remark (probability leakage). Truncating the KL to $\mathcal{S}_t^k$ may in principle encourage shifting probability mass into the ignored tail. As a simple safeguard, we recommend always including the sampled token, i.e., $\tilde{\mathcal{S}}_t^k = \mathcal{S}_t^k \cup \{o_t\}$. In practice, under the Zipfian next-token distribution, o_t already falls inside $\mathcal{S}_t^k$ with high probability when the Top-k mass is near 1.

4.4 Variance Reduction via Rao-Blackwellization

Standard KL estimators introduce stochasticity from two sources: the sampling of the trajectory/history h_t and the sampling of a token-level random variable. TIKE reduces the latter by analytically summing over $\mathcal{S}_t^k$.

A token-sampling baseline (for variance comparison). Fix a history h_t and consider the following (conceptual) single-sample estimator that uses an *additional* token draw $V_t \sim \pi_\theta(\cdot \mid h_t)$:

$$D_t^{\text{MC-TopK}} \triangleq r_t^{IS} \cdot \mathbf{1}\{V_t \in \mathcal{S}_t^k\} \cdot \log \frac{\pi_\theta(V_t \mid h_t)}{\pi_{\text{ref}}(V_t \mid h_t)}. \tag{15}$$

Conditioned on (h_t, o_t), r_t^{IS} is fixed and the only remaining randomness comes from V_t.

Rao–Blackwellization.

$$\begin{aligned}\text{Var}\left(D_t^{\text{TopK}}\right) &= \text{Var}\left(\mathbb{E}[D_t^{\text{MC-TopK}} \mid h_t, o_t]\right) \\ &\leq \text{Var}\left(D_t^{\text{MC-TopK}}\right). \end{aligned} \tag{16}$$

Intuitively, TIKE eliminates the high-variance "tail-hit" behavior from token-level Monte Carlo sampling by integrating the Top-k components analytically. This aligns with Rao–Blackwellized KL estimators studied in Amini et al. (2025), while our setting additionally imposes Top-k truncation for memory efficiency.

4.5 Gradient Estimation and Stability

With $\text{sg}(\cdot)$ denoting stop-gradient, the KL penalty used in optimization is D_t^{TopK}. Therefore its gradient is tractable:

$$\begin{aligned}\nabla_\theta D_t^{\text{TopK}} = \text{sg}(r_t^{IS}) \sum_{v\in\mathcal{S}_t^k} \\ \nabla_\theta\left(\pi_\theta(v \mid h_t) \log \frac{\pi_\theta(v \mid h_t)}{\pi_{\text{ref}}(v \mid h_t)}\right). \end{aligned} \tag{17}$$

Properly scoped unbiasedness. Because the summation is analytic over $\mathcal{S}_t^k$, the gradient above is the exact gradient of the *truncated* objective (at fixed h_t, under the stop-gradient convention on the Top-k index selection). This does not remove the state-distribution shift incurred by sampling h_t from $d^{\pi_{\text{old}}}$, but empirically GRPO-style clipping keeps π_θ close to π_{old}, mitigating the discrepancy (Shao et al., 2024).

5 Experiments

To evaluate the effectiveness of the proposed TIKE, we conduct experiments using the Qwen2.5 series (Team et al., 2024), specifically Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct, on a suite of mathematical reasoning benchmarks including AIME24, AIME25 (Mathematical Association of America, 2025), MATH500 (Lightman et al., 2023), and Minerva (Lewkowycz et al., 2022).

Experiment Setup. The training is implemented via the *verl* framework (Sheng et al., 2025) using Group Relative Policy Optimization (GRPO) (Shao et al., 2024) on the DAPO-Math-17k dataset (Yu et al., 2025). We standardize the training configuration across all runs: each model is trained for 500 steps with a global batch size of 512 and a KL regularization coefficient $\beta = 0.001$, following the default setting in *verl*. For hardware, we utilize 8 NVIDIA H200 GPUs (141GB memory each) for the 7B model and 4 NVIDIA H200 GPUs for the 1.5B model. During training, we sample $G = 8$ rollouts per prompt with a maximum response length of 4096 tokens to support extended reasoning chains. The optimization is performed using AdamW (Kingma, 2014) with a learning rate of 1×10^{-6}. Performance is evaluated using Pass@n and Mean@n metrics. For the AIME24 and AIME25 datasets, which consist of only 30 problems each, we sample 32 responses per question to report Pass@32 and Mean@32; for the MATH500 and Minerva datasets, which contain hundreds of problems, we sample 4 responses per question to report Pass@4 and Mean@4.

Method	AIME24		AIME25		MATH500		Minerva	
	Pass@32	Mean@32	Pass@32	Mean@32	Pass@4	Mean@4	Pass@4	Mean@4
Qwen2.5-7B-Instruct, Max Response Len=4K								
Base	13.33	6.25	13.95	2.92	55.25	52.98	21.84	20.04
K1	$28.65_{+15.3}$	$18.13_{+11.9}$	$23.26_{+9.3}$	$15.00_{+12.1}$	$68.24_{+13.0}$	$65.40_{+12.4}$	$24.32_{+2.5}$	$22.55_{+2.5}$
K2	$26.52_{+13.2}$	$19.52_{+13.3}$	$23.24_{+9.3}$	$13.33_{+10.4}$	$68.48_{+13.2}$	$66.35_{+13.4}$	$25.19_{+3.4}$	$23.44_{+3.4}$
K3	$29.16_{+15.8}$	$18.75_{+12.5}$	$24.18_{+10.2}$	$18.85_{+15.9}$	$68.65_{+13.4}$	$65.30_{+12.3}$	$24.69_{+2.9}$	$22.61_{+2.6}$
TIKE$_8$	$\mathbf{30.00}_{+16.7}$	$20.00_{+13.8}$	$\mathbf{29.97}_{+16.0}$	$\mathbf{18.96}_{+16.0}$	$\mathbf{69.14}_{+13.9}$	$\mathbf{66.85}_{+13.9}$	$25.27_{+3.4}$	$22.98_{+2.9}$
TIKE$_{16}$	$28.10_{+14.8}$	$\mathbf{21.25}_{+15.0}$	$26.19_{+12.2}$	$16.77_{+13.9}$	$67.71_{+12.5}$	$65.45_{+12.5}$	$\mathbf{26.17}_{+4.3}$	$\mathbf{23.71}_{+3.7}$
Qwen2.5-1.5B-Instruct, Max Response Len=4K								
Base	5.34	4.17	2.12	0.10	41.53	35.75	5.43	4.32
K1	$\mathbf{18.08}_{+12.7}$	$7.71_{+3.5}$	$10.53_{+8.4}$	$3.96_{+3.9}$	$51.55_{+10.0}$	$48.35_{+12.6}$	$12.76_{+7.3}$	$10.94_{+6.6}$
K2	$13.85_{+8.5}$	$7.92_{+3.8}$	$8.25_{+6.1}$	$1.35_{+1.3}$	$51.41_{+9.9}$	$46.55_{+10.8}$	$13.33_{+7.9}$	$11.31_{+7.0}$
K3	$15.42_{+10.1}$	$6.98_{+2.8}$	$12.68_{+10.6}$	$2.40_{+2.3}$	$51.60_{+10.1}$	$47.35_{+11.6}$	$12.21_{+6.8}$	$10.85_{+6.5}$
TIKE$_8$	$17.72_{+12.4}$	$6.56_{+2.4}$	$11.88_{+9.8}$	$\mathbf{4.06}_{+4.0}$	$51.77_{+10.2}$	$46.90_{+11.2}$	$11.46_{+6.0}$	$9.93_{+5.6}$
TIKE$_{16}$	$16.22_{+10.9}$	$\mathbf{8.65}_{+4.5}$	$\mathbf{14.20}_{+12.1}$	$\mathbf{4.06}_{+4.0}$	$\mathbf{54.16}_{+12.6}$	$\mathbf{49.60}_{+13.9}$	$\mathbf{15.01}_{+9.6}$	$\mathbf{12.32}_{+8.0}$

Table 1: Comparison of different KL estimators on reasoning benchmarks.

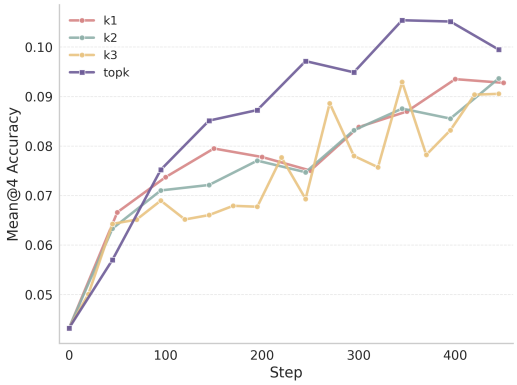

Figure 3: Mean@4 accuracy on Minerva validation set with Qwen2.5-1.5B-Instruct. Curves are smoothed with 0.9 EMA; "topk" denotes the $k = 16$ estimator.

Result Analysis. The main experimental results, summarized in Table 1, demonstrate that the proposed TIKE consistently outperforms traditional stochastic baselines across all evaluated benchmarks and model scales. Specifically, for Qwen2.5-7B-Instruct, the TIKE$_8$ estimator achieves a significant Pass@32 score of 30.00 on AIME24, representing a +16.7 absolute improvement over the base model. Similarly, on the 1.5B model, TIKE$_{16}$ yields the highest performance on MATH500 and Minerva, reaching 54.16 and 15.01 Pass@4, respectively. The training dynamics on the Minerva validation set for Qwen2.5-1.5B-Instruct are illustrated in Figure 3. It is evident that TIKE results in a much faster and more sustained ascent in accuracy compared to other methods. This performance gain is further supported by the fact that the Top-k estimator exhibits a more stable improvement in Mean@4 accuracy compared to the fluctuating trajectories of stochastic estimators. These results suggest that by truncating the long tail of the distribution and focusing on the most significant probability mass, our method effectively reduces gradient noise and provides a more reliable regularization signal for reinforcement learning in mathematical reasoning tasks.

5.1 Variance Analysis

To rigorously evaluate the bias-variance trade-off of the proposed estimator compared to existing baselines, we conducted a controlled simulation focusing on the stability of the gradient estimates.

Experimental Setup. We employed the Qwen2.5-0.5B-Instruct model as the reference policy π_{ref}. To obtain a policy model π_θ that has deviated from the reference but retains reasoning capabilities, we fine-tuned π_{ref} on the GSM8K dataset (Cobbe et al., 2021) using Group Relative Policy Optimization (GRPO). The training was

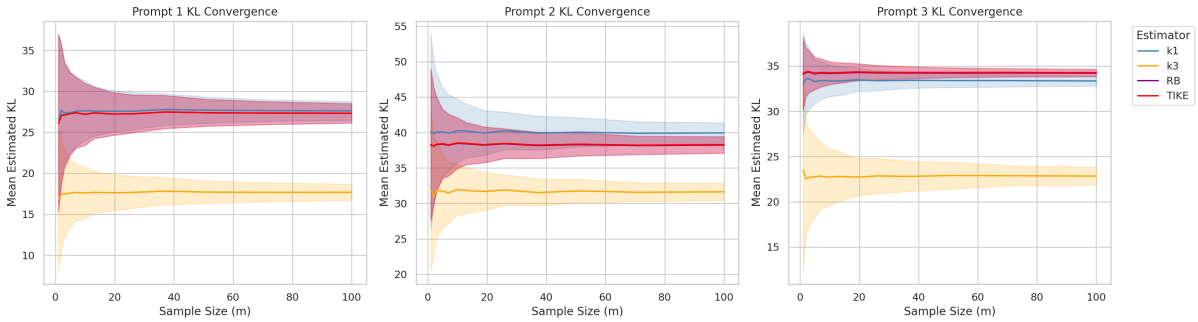

Figure 4: **Bias-Variance Analysis of KL Estimators.** Convergence of sequence-level KL estimates as the sample size M increases. The analysis is performed on three distinct prompts from GSM8K.

conducted for 3,000 steps with a learning rate of 1×10^{-6}, a global batch size of 128, and a maximum completion length of 2,048 tokens. This setup ensures that π_θ and π_{ref} have a non-trivial KL divergence, simulating a realistic mid-training scenario.

Data Generation and Bootstrap Estimation. We selected three diverse mathematical prompts from the GSM8K dataset to serve as the basis for evaluation. For each prompt x, we generated a large pool of $N_{\text{pool}} = 2,000$ response sequences y using the trained policy π_θ.

To analyze the convergence properties of the estimators, we performed a bootstrap analysis. Let $\hat{J}$ denote a generic KL estimator. For a range of sample sizes M (spaced logarithmically), we drew M sequences from the generated pool with replacement. We repeated this resampling process $B = 1000$ times for each M. We report the expected KL estimate and the empirical standard deviation across these bootstrap trials. We compared four estimators: the standard k_1 estimator, the Control Variate estimator with $\alpha = 1$ (k_3), the Rao-Blackwellized estimator (RB), which is also the exact expectation of KL, and our proposed TIKE.

Results Analysis. The simulation results, illustrated in Figure 4, reveal critical insights regarding the fidelity and stability of the estimators.

First, regarding estimation accuracy, the **RB** estimator (purple) serves as the ground truth, representing the exact expected KL divergence calculated via full-vocabulary integration. We observe that our proposed **TIKE** (red) consistently tracks the RB baseline with high fidelity across all three prompts. In contrast, the stochastic Monte Carlo estimators (k_1 and k_3) often exhibit deviations from this exact expectation. For instance, in Prompts 1 and 3, the k_3 estimator converges to a value substantially lower than the true KL.

Second, regarding variance, **TIKE** demonstrates superior consistency. As indicated by the shaded confidence regions, the k_3 estimator exhibits variable behavior, especially at small sample sizes ($M < 20$). The k_1 estimator shows a standard variance decay but remains noisier than the dense methods. **TIKE**, conversely, maintains a consistently narrow variance profile across all prompts. It is reliably smaller than that of k_3 and k_1, providing a more stable gradient signal closer to the deterministic RB baseline.

5.2 Additional Analyses

We provide extensive supplementary experiments in the Appendix to further validate TIKE's robustness and practical applicability. Specifically: (1) **Top-k coverage analysis** (Appendix A.1): Truncation residuals are negligible in high-mass regimes and manageable elsewhere, with diminishing returns beyond k=16. (2) **Cross-architecture generalization** (Appendix A.2): TIKE$_{32}$ consistently outperforms K3 on DeepSeek-R1-Distill-Llama-8B, confirming the benefit transfers beyond the Qwen family. (3) **Computational cost** (Appendix A.3): At k=16, overhead is minimal ($+6.9\%$ step time, $+0.1\%$ memory), while full-vocabulary integration is prohibitive. (4) **KL coefficient robustness** (Appendix A.4): TIKE outperforms K3 under different β values, demonstrating robustness to this hyperparameter. (5) **Multi-seed reproducibility** (Appendix A.5): Paired comparisons across multiple seeds with matched random states show consistent improvements. In summary, the supplementary experiments further reinforce the conclusions drawn in the previous sections.

6 Conclusion

In this work, we addressed the critical instability inherent in Reinforcement Learning for Large Language Models, specifically identifying the variance-bias trade-off in KL divergence estimation. We proposed **TIKE** (Top-k Importance-weighted KL Estimator), a method that theoretically and empirically bridges the gap between the high variance of Monte Carlo sampling and the prohibitive memory costs of exact Rao-Blackwellization. TIKE computes a deterministic partial expectation over the dominant probability mass, effectively filtering out tail noise. Our theoretical analysis confirms that TIKE provides a low-variance approximation to the true KL objective, and our variance analysis demonstrates that it closely tracks the exact Rao-Blackwellized estimator with substantially reduced gradient noise. Empirically, extensive experiments using the Qwen2.5 series (1.5B and 7B) and DeepSeek-R1-Distill-Llama-8B on complex mathematical reasoning benchmarks—including AIME24, AIME25, MATH500, and Minerva—demonstrate that TIKE consistently outperforms standard stochastic estimators (k_1, k_2, k_3). By achieving near-exact KL estimation at a fraction of the memory cost, TIKE offers a practical and principled solution for scaling reinforcement learning to increasingly capable language models.

Limitations

While TIKE demonstrates robust performance across mathematical reasoning benchmarks, several limitations merit consideration. First, its effectiveness relies on the Zipfian sparsity hypothesis—the assumption that probability mass concentrates heavily in the top-k tokens. In high-entropy generation scenarios, such as open-ended creative writing or diverse dialogue generation, heavier distributional tails may lead to non-negligible truncation bias. Second, although significantly more memory-efficient than full Rao-Blackwellization, the required `TopK` operation introduces slight computational overhead compared to single-sample estimators, which may impact throughput in extremely large-scale training. Third, the hyperparameter k requires task-specific tuning: while $k \in \{8, 16\}$ proved effective across our benchmarks, optimal values may vary with model architecture, vocabulary size, and task characteristics. Finally, while we have validated TIKE across both the Qwen and Llama model families (Appendix A.2), our empirical evaluation focused on mathematical reasoning; generalization to other domains such as code generation, instruction following, or multi-turn dialogue remains to be validated.

Ethics Statement

TIKE is designed to improve the stability and efficiency of reinforcement learning for large language models, and we encourage practitioners to pair such methods with robust safety evaluations and alignment procedures. All experiments follow relevant data usage policies and licensing terms: the datasets employedDAPO-Math-17k for training, and AIME24, AIME25, MATH500, Minerva, and GSM8K for evaluationare publicly available benchmarks, and the Qwen2.5 model series (0.5B, 1.5B, and 7B variants) used in our experiments is released under open-source licenses permitting research use. To facilitate reproducibility, our implementation code is available at a GitHub repository. Our experiments were conducted on NVIDIA H200 GPUs; while large-scale training incurs substantial energy consumption, TIKE's memory efficiency may contribute to reducing computational waste. Finally, we used Cursor to assist with code development in accordance with ACL submission policies and broader research ethics guidelines, while all scientific contributions, experimental design, and manuscript content reflect the authors' original work.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. U24B20181 and 62525602).

References

Afra Amini, Tim Vieira, and Ryan Cotterell. 2025. Better estimation of the kullback–leibler divergence between language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

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

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang

He, Yuchen Fan, Tianyu Yu, and 1 others. 2025. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

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

Riccardo Di Sipio. 2025. Rethinking llm training through information geometry and quantum metrics. *arXiv preprint arXiv:2506.15830*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jian Hu. 2025. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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.

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*.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Keliang Liu, Dingkang Yang, Ziyun Qian, Weijie Yin, Yuchi Wang, Hongsheng Li, Jun Liu, Peng Zhai, Yang Liu, and Lihua Zhang. 2025b. Reinforcement learning meets large language models: A survey of advancements and applications across the llm lifecycle. *arXiv preprint arXiv:2509.16679*.

Kezhao Liu, Jason Klein Liu, Mingtao Chen, and Yiming Liu. 2025c. Rethinking kl regularization in rlhf: From value estimation to gradient optimization. *arXiv preprint arXiv:2510.01555*.

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

Chengqi Lyu, Songyang Gao, Yuzhe Gu, Wenwei Zhang, Jianfei Gao, Kuikun Liu, Ziyi Wang, Shuaibin Li, Qian Zhao, Haian Huang, and 1 others. 2025. Exploring the limit of outcome reward for learning mathematical reasoning. *arXiv preprint arXiv:2502.06781*.

Mathematical Association of America. 2025. American invitational mathematics examination (AIME). https://maa.org/. Accessed: 2025-01-05.

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.

Sheldon M. Ross. 2002. *Simulation*, 3rd edition. Academic Press.

John Schulman. 2020. Approximating KL divergence. http://joschu.net/blog/kl-approx.html. Accessed: 2026-01-05.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.

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

Vedant Shah, Johan Obando-Ceron, Vineet Jain, Brian Bartoldson, Bhavya Kailkhura, Sarthak Mittal, Glen Berseth, Pablo Samuel Castro, Yoshua Bengio, Nikolay Malkin, and 1 others. 2025. A comedy of estimators: On kl regularization in rl training of llms. *arXiv preprint arXiv:2512.21852*.

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*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021.

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*.

Qwen Team and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3).

Catriona Tullo and James Hurford. 2003. Modelling zipfian distributions in language. In *Proceedings of language evolution and computation workshop/course at ESSLLI*, pages 62–75.

Tim Van Erven and Peter Harremos. 2014. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, and 1 others. 2025. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. 2023. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. *arXiv preprint arXiv:2312.11456*.

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*.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, and 1 others. 2025. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*.

Yifan Zhang, Yifeng Liu, Huizhuo Yuan, Yang Yuan, Quanquan Gu, and Andrew Chi-Chih Yao. 2025. On the design of kl-regularized policy gradient algorithms for llm reasoning. *arXiv preprint arXiv:2505.17508*.

Elizaveta Zhemchuzhina, Nikolai Filippov, and Ivan P Yamshchikov. 2022. Pragmatic constraint on distributional semantics. *arXiv preprint arXiv:2211.11041*.

Congming Zheng, Jiachen Zhu, Zhuoying Ou, Yuxiang Chen, Kangning Zhang, Rong Shan, Zeyu Zheng, Mengyue Yang, Jianghao Lin, Yong Yu, and 1 others. 2025. A survey of process reward models: From outcome signals to process supervisions for large language models. *arXiv preprint arXiv:2510.08049*.

A Appendix

A.1 Top-k Coverage and Truncation Residual Analysis

Step	Regime	Frac.	mean m_{16}	mean $H(\pi_\theta)$	mean $\lvert\xi_{\text{tail}}\rvert$	p95 $\lvert\xi_{\text{tail}}\rvert$
10	$m_{16}\geq 0.99$	0.332	0.9983	—	1.68e-4	7.12e-4
	$m_{16}<0.90$	—	0.709	3.94	1.30e-2	3.34e-2
50	$m_{16}\geq 0.99$	0.347	0.9986	—	5.07e-4	2.24e-3
	$m_{16}<0.90$	—	0.650	4.45	2.11e-2	5.08e-2
90	$m_{16}\geq 0.99$	0.387	0.9987	—	6.45e-4	3.07e-3
	$m_{16}<0.90$	—	0.690	4.10	2.59e-2	5.89e-2

Table 2: Truncation residual stratified by Top-16 mass coverage regime at different training steps. "Frac." denotes the fraction of token positions falling in the high-mass bucket; entropy $H(\pi_\theta)$ is reported only for the low-mass regime to characterize its distributional spread.

We audit the truncation residual $\xi_{\text{tail}}(h_t) = D_{\text{KL}}(h_t) - D^k_{\text{KL}}(h_t)$ and the Top-k mass coverage $m_k(h_t) = \sum_{v\in\mathcal{S}^k_t} \pi_\theta(v \mid h_t)$ on Qwen2.5-1.5B-Instruct TIKE checkpoints at training steps 10, 50, and 90. We use 32 prompts from DAPO-Math-17k, generating 512 tokens per prompt (temperature $= 1.0$), and evaluate $\sim$30K token-positions per checkpoint. At each position, we compute both the exact full-vocabulary KL and the truncated Top-k KL.

Residual by Mass Regime Table 2 reports token-level statistics of the truncation residual, stratified by the Top-16 mass coverage m_{16} into a high-mass regime ($m_{16} \geq 0.99$) and a low-mass regime ($m_{16} < 0.90$). In the high-mass regime, the residual is negligible (p95 $|\xi_{\text{tail}}| \leq 3.1 \times 10^{-3}$ for k=16). In the low-mass regime, which typically corresponds to higher-entropy token positions, residuals increase but remain within a manageable range.

A.2 Cross-Architecture Generalization

To verify that TIKE's benefits are not limited to the Qwen model family, we evaluate on a different backbone architecture: DeepSeek-R1-Distill-Llama-8B (Liu et al., 2025a). We compare TIKE_{32} against the K3 baseline under the same evaluation protocol used in the main experiments. Table 3 reports the results.

The results demonstrate consistent gains of TIKE over K3 across all benchmarks and metrics. The improvements are particularly pronounced on

Method	AIME25		AMC23		AIME24	
	Mean	Pass	Mean	Pass	Mean	Pass
K3	26.25	44.95	79.53	93.82	32.19	70.93
TIKE$_{32}$	**30.31**	**53.23**	**87.11**	**94.66**	**41.56**	**74.52**
Δ	+4.06	+8.28	+7.58	+0.84	+9.37	+3.59

Table 3: TIKE$_{32}$ vs. K3 on DeepSeek-R1-Distill-Llama-8B across reasoning benchmarks. TIKE consistently outperforms K3 on all metrics (Mean@32, Pass@32), confirming the benefit transfers beyond the Qwen family.

AIME24 (+9.37 Mean@32) and AIME25 (+8.28 Pass@32), confirming that the benefit of TIKE-based KL estimation generalizes across model families.

A.3 Computational Cost Profiling

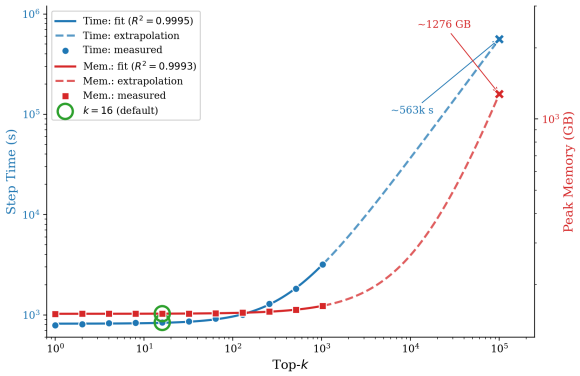

Figure 5: Computational cost of TIKE as a function of Top-k on DeepSeek-R1-Distill-Llama-8B (log–log axes, dual y-axis). Markers are measured; solid lines are power-law fits; dashed segments extrapolate to full vocabulary ($|\mathcal{V}| \approx 100$K). The green circle at k=16 shows negligible overhead over the `k3/top1` baseline.

To quantify the computational overhead of TIKE, we instrumented training with `torch.profiler` and recorded per-step wall-clock time and peak GPU memory for each Top-k setting on DeepSeek-R1-Distill-Llama-8B, using `k3/top1` as the baseline. Each configuration was run repeatedly until convergence behavior stabilized, and the main training sweep covered k up to 512; larger values were additionally profiled to characterize scaling behavior. Figure 5 visualizes the results on log–log axes, and we fit a power-law model $y = a + b \cdot k^p$ to the measurements, obtaining $\mathrm{Time}(k) = 816.44 + 0.5998 \cdot k^{1.194}$ ($R^2 = 0.99954$, RMSE $= 14.90$) and

Setting	Time (s)	ΔTime (s)	Mem. (GB)	ΔMem. (GB)
k3/top1	785.6	—	150.5	—
top16†	840.1	+54.5	150.7	+0.2
top1024‡	3185.0	+2399.4	162.7	+12.2
*top100k**	*∼563k*	*+562k*	*∼1276*	*+1125*

Table 4: Cost at key operating points (full scaling in Figure 5). † recommended default; ‡ profiled maximum; * power-law extrapolation to full vocabulary.

$\mathrm{Mem}(k) = 150.54 + 0.01289 \cdot k^{0.988}$ ($R^2 = 0.99926$, RMSE $= 0.0984$). These fits indicate that step time grows super-linearly ($p \approx 1.19$) while memory grows near-linearly ($p \approx 0.99$) in k over the profiled range, explaining why very large k quickly becomes impractical: extrapolating to a full vocabulary such as Qwen's ($|\mathcal{V}| \approx 100$K) yields a prohibitive step time of ∼563k s and ∼1276 GB peak memory—neither is tolerable for RL training.

Table 4 reports the cost at four representative operating points. At the recommended default k=16, step time increases by only +54.5 s (+6.9%) and peak memory by only +0.2 GB (+0.1%) relative to the `k3/top1` single-sample baseline, i.e., nearly the same resource footprint in practice. These results confirm that TIKE occupies a favorable regime on the accuracy–cost frontier: it retains the fidelity of dense integration on the Top-k support while avoiding the prohibitive cost of full-vocabulary KL computation.

A.4 KL Coefficient Robustness

β	Method	AIME25		AMC23	
		Mean	Pass	Mean	Pass
1E-3	TIKE	**3.54**	**14.10**	**39.45**	**58.96**
	K3	1.67	6.53	35.00	53.08
5E-4	TIKE	1.89	**9.15**	**35.70**	**56.38**
	K3	**2.92**	7.58	34.70	52.22

Table 5: TIKE vs. K3 under different KL coefficients (β). TIKE remains competitive or superior across both settings (Mean@32, Pass@32), demonstrating robustness to this hyperparameter.

To verify that TIKE's advantage is not tied to a single KL coefficient, we compare TIKE and K3 under two values of β on Qwen2.5-1.5B-Instruct with 200 GRPO training steps. As shown in Ta-

Group	Method	AIME25		AMC23		AIME24	
		mean@32	pass@32	mean@32	pass@32	mean@32	pass@32
Run 1	TIKE_{16}	3.54	14.10	39.45	58.96	9.17	16.20
	K3	1.67	6.53	35.00	53.08	6.35	14.06
Run 2	TIKE_{16}	2.29	10.49	38.00	57.99	7.92	14.14
	K3	1.88	9.67	37.08	56.99	7.17	15.26
Run 3	TIKE_{16}	2.92	6.63	39.67	64.77	8.23	16.60
	K3	1.98	8.65	32.37	47.30	6.71	12.80
Run 4	TIKE_{16}	3.44	11.47	35.31	55.14	7.81	14.98
	K3	2.60	10.68	33.83	52.18	6.98	14.99
Mean $\pm$ SD	TIKE_{16}	3.05 ± 0.57	10.67 ± 3.10	38.11 ± 2.01	59.22 ± 4.04	8.28 ± 0.62	15.48 ± 1.13
	K3	2.03 ± 0.40	8.88 ± 1.77	34.57 ± 1.99	52.39 ± 3.98	6.80 ± 0.36	14.28 ± 1.11
δ_{Mean}	TIKE_{16} − K3	1.02	1.79	3.54	6.83	1.48	1.20
t	paired one-sided	3.303	0.878	2.400	1.852	3.090	1.094
p	paired one-sided	**0.023**	0.222	**0.048**	0.081	**0.027**	0.177

Table 6: Multi-seed reproducibility results for TIKE_{16} vs. K3 on Qwen2.5-1.5B-Instruct (200 steps, 4 seeds). The table consolidates per-seed results, aggregated statistics (mean $\pm$ SD), and paired one-sided t-tests (H_1 : $\mu_{\text{TIKE}-\text{K3}} > 0$). Significant p-values ($p < 0.05$) are bolded.

ble 5, TIKE outperforms K3 on most metrics under both coefficients, indicating that the observed advantage is not an artifact of a specific β setting. The consistent improvement across $\beta \in \{0.001, 0.0005\}$ supports the general applicability of TIKE.

k	**mean** m_k	**mean** $\lvert\xi_{\text{tail}}\rvert$	**p95** $\lvert\xi_{\text{tail}}\rvert$
16	0.876	1.24e-2	4.11e-2
64	0.921	8.08e-3	2.72e-2
256	0.948	5.20e-3	1.71e-2

Table 7: Truncation residual at step 90 for varying k. Larger k reduces the residual, but with diminishing marginal benefit beyond k=16.

Effect of Increasing *k* Table 7 shows that the residual diminishes monotonically as k increases, with diminishing returns beyond k=16 in our setting (reported at step 90).

These results suggest that k=16 provides a practical default: residuals are negligible in the high-mass regime (which covers a substantial fraction of token positions), and remain manageable even in higher-entropy regions. Practitioners may increase k when task-specific error tolerance demands tighter approximation.

A.5 Multi-Seed Reproducibility

To assess reproducibility and statistical reliability, we conduct a paired multi-seed comparison between TIKE_{16} and K3 using Qwen2.5-1.5B-Instruct with GRPO for 200 training steps. We use 4 independent seeds with matched random states, and Table 6 consolidates the evidence: the upper block lists per-seed results, the middle block aggregates them into mean $\pm$ SD, and the lower block reports the paired mean difference together with one-sided t-tests under $H_1 : \mu_{\text{TIKE}-\text{K3}} > 0$.

Three observations stand out. First, TIKE_{16} achieves a positive mean improvement over K3 on all six metrics, with the largest absolute gains on AMC23 ($+3.54$ mean@32 and $+6.83$ pass@32). Second, the improvements are statistically significant ($p < 0.05$) on three mean@32 metrics (AIME25, AMC23, AIME24); the remaining three metrics, while not reaching significance at this sample size, trend in the same direction with positive t-statistics. Third, the per-seed rows show that TIKE_{16} outperforms K3 on the majority of metric–seed cells rather than being driven by a single favorable run, consistent with the paired test design. We acknowledge that four seeds provide limited statistical power; nevertheless, the consistent direction of effect across all metrics, combined with the variance reduction from pairing, supports the robustness of the observed gains.