ENABLING REPARAMETERIZED POLICY GRADIENTS FOR LLM POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

The score function (SF) estimator serves as the basis for nearly all reinforcement learning (RL) pipelines currently used to finetune Large Language Models (LLMs). While general, these methods rely exclusively on samples to derive an estimate of the policy gradient, which leads to a limited ability to explore beyond a base policy's distribution. Conversely, one could elect to directly differentiate through a reparameterized objective to fully leverage a reward model's gradient towards improvement rather than estimating it through samples. That said, reparameterized policy gradients have yet to be explored in this area because of the added complexity of differentiating through a discrete sampling objective. In addition, these types of gradients have historically been avoided by RL practitioners, due to their instability and impracticality in long horizon tasks involving autoregressive generation. In this paper, we challenge these concerns and provide a principled framework for efficiently computing the reparameterized policy gradient for finetuning LLMs with RL. In a series of experiments spanning both preference alignment and reasoning, we show that directly backpropagating through a relaxed objective is not only feasible, but can actually be competitive with current state of the art SF-based estimators despite the literature's warnings against it. Finally, we highlight the unique property of reparameterized policy gradients to explore novel solutions outside the initial sampling distribution.

1 INTRODUCTION

Reinforcement learning (RL) has emerged as the dominant paradigm for finetuning Large Language Models (LLMs) beyond next token prediction. Whether rewards are derived from human preferences or obtained through some ground truth verifier, RL algorithms have empowered modern LLMs with impressive reasoning and problem solving capabilities (Shao et al., 2024; Yu et al., 2025) while remaining aligned with human preferences (Ouyang et al., 2022).

Current approaches for LLM post-training such as Group Relative Policy Optimization (GRPO) (Shao et al., 2024), are built on top of the score function estimator (Williams, 1992) which provides Monte Carlo estimates of a reward's gradients with respect to the parameters of a language model, also known as the policy gradient. These methods are particularly appealing due to their generality; they make no assumptions on the structure of a reward function, instead treating it as a black box function over any given state or completion. While traditional failure modes of the score function estimator such as high variance (Sutton et al.) have not been an issue for language modeling (Ahmadian et al., 2024; Kazemnejad et al., 2025), its reliance on samples has been shown to limit RL's ability to find novel solutions outside the base policy's original support (Yue et al., 2025; Wu et al., 2026).

Instead of performing a zero-th order estimate of the policy gradient, another popular approach in RL is to transform the policy's objective to be differentiable end-to-end through a combination of model learning and reparameterization, allowing us to directly compute its gradient with standard backpropagation. This class of methods is sometimes referred to as backpropagation-based policy optimization (BPO) (Heess et al., 2015), reparameterized policy gradients (Deisenroth & Rasmussen, 2011), pathwise derivatives (Clavera et al., 2020; Hafner et al., 2021), or first order policy gradients (Suh et al., 2022) in model-based RL. With its rich history in the literature, BPO often

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.

promises to be a lower variance estimator or offer improved sample efficiency by directly leveraging the gradient of a given world model.

Despite their potential advantages, reparameterized policy gradients have yet to be explored in the context of RL for finetuning language models. The discrete nature of the text generation setting presents several discontinuities in the reward landscape, making direct differentiation impossible without additional approximations. Moreover, even when possible, these methods usually scale poorly with an environment's horizon, because the accumulation of auto-regressive gradients is computationally expensive (Williams & Zipser, 1989), theoretically unstable (Parmas et al., 2023; Ma et al., 2024; Metz et al., 2022; Parmas et al., 2018), and practically discouraged (Hafner et al., 2022; 2023; Heess et al., 2015). In this work, we challenge several of these shortcomings as well as novel ones unique to this setup in order to enable reparameterized policy gradients for LLM post-training.

We introduce **T**erminal **A**ware **G**umbel **BPO**, a novel first order policy gradient for finetuning LLMs. TAG-BPO primarily uses a Gumbel Softmax estimator and the reparameterization trick (Jang et al., 2016) to resolve the discontinuities from sampling discrete tokens. However, we show that this is not sufficient and that the terminal condition requires additional special attention in language modeling to make the policy gradient exact. In addition, we suggest an alternative Kullback-Leibler (KL) loss to mitigate the gradient bias induced by current KL penalties (Shah et al., 2026) which are particularly harmful to reparameterized policy gradients. On the axis of practicality, the gradients from TAG-BPO are decoupled from generation, allowing them to be computed with a single batched forward pass using samples taken from any generation process, just like GRPO. Theoretically, we demonstrate why traditional concerns about exploding gradients when backpropagating through long sequences do not apply in this setting by extending the results from Ma et al. (2024). Finally, we perform experiments in both Reinforcement Learning from Human Feedback (RLHF) and Reinforcement Learning from Verifiable Rewards (RLVR) environments to establish a baseline performance of these types of gradients for LLM post-training. While TAG-BPO can perform similarly to current state-of-the-art score function based algorithms, the implicit anchoring of score function methods such as GRPO to the base policy can act as a useful constraint, limiting reward over-optimization even though it also restricts exploration.

2 Related Works

RL for LLM post-training. RL for LLM post-training has converged on progressively simpler on-policy algorithms across both RLHF and RLVR. PPO (Schulman et al., 2017), introduced alongside RLHF for LMs (Ziegler et al., 2020; Stiennon et al., 2022) and scaled to instruction following (Ouyang et al., 2022; Bai et al., 2022), has largely given way to group-baseline policy gradient methods (Ahmadian et al., 2024; Shao et al., 2024), which now underpin both RLHF pipelines and large-scale training against verifiable rewards (Lambert et al., 2025; DeepSeek-AI et al., 2025; Yu et al., 2025; Liu et al., 2025b). Despite differing in how they form baselines, normalize advantages, and clip updates, these methods remain score function estimators without exception. Empirically, these methods have a tendency to remain close to and sharpen the pretrained model's existing behavior instead of expanding upon it (Kirk et al., 2024; Yue et al., 2025; Wu et al., 2026; Cui et al., 2025). Recent work counteracts these limitations through prolonged or restructured RL training (Liu et al., 2025a), explicit entropy or support interventions (Wu et al., 2026), and curriculum-based exploration schedules (Cai et al., 2026), but each remains a score function estimator.

Model-based RL and reparameterized gradients. The estimator we adopt has a long history in RL, where policy gradients can be obtained by differentiating through a model of the environment rather than through sampled trajectories: from analytic gradients through learned dynamics (Deisenroth & Rasmussen, 2011; Heess et al., 2015) to actor-critic methods that backpropagate through learned latent world models (Clavera et al., 2020; Hafner et al., 2023; Ghugare et al., 2023). The central obstacle is that, over long horizons, gradients must propagate recursively through the learned dynamics, producing products of Jacobians that can lead to exploding or vanishing gradients, high estimator variance, and increasingly ill-conditioned optimization landscapes (Parmas et al., 2018; Metz et al., 2022), with additional bias arising when the objective is discontinuous (Suh et al., 2022). Existing remedies trade stability for bias or added machinery, either truncating the differentiated horizon (Xu et al., 2022; Georgiev et al., 2024) or combining pathwise and score

	Challenges in using RP policy gradients	Research area and references	Addressed in this work
Existing	Prohibitive costs of accumulating gradients during auto-regressive generation	Fine-tuning diffusion models (Clark et al., 2024; Zekri & Boullé, 2025), Sequence modeling (Williams & Zipser, 1989; Werbos, 1990; Gruslys et al., 2016)	Section 4.2 ✓
	Unstable gradients from Jacobian products over auto-regressive trajectories	Model-based RL (Parmas et al., 2018; 2023; Suh et al., 2022; Ma et al., 2024), Sequence modeling (Pascanu et al., 2013)	Section 4.4 ✓
	Bias from a soft relaxation of a discrete variable	Model-based RL (Hafner et al., 2021), Supervised Learning (Jang et al., 2016)	✗
	Instabilities due to the curvature of the reward model's gradients	GANs (Arjovsky et al., 2017; Gulrajani et al., 2017), Adversarial attacks for LLMs (Ebrahimi et al., 2018; Zou et al., 2023; Wichers et al., 2024)	✗
Novel	Discrete terminal condition	LLM post-training with RL (this work)	Section 4.1 ✓
	`K3` estimator gradient bias	LLM post-training with RL (this work)	Section 4.3✓

Table 1: Challenges unique to using reparameterized policy gradients for finetuning LLMs instead of using a score function estimator taken from various other research areas. The references are not exhaustive, and aim to represent only the most notable works within a given research area and challenge.

function estimators under inverse-variance weights (Parmas et al., 2023). Ma et al. (2024) attribute these long-horizon pathologies to the autoregressive dependence on the model's own state predictions, showing that world models predicting future states directly from the initial state and action sequence yield polynomial rather than exponential gradient norm growth.

Reparameterized gradients in language modeling. Continuous relaxations of discrete token sampling have been used to propagate scoring model gradients directly into text generators, from per-token temperature annealing (Hu et al., 2018) to text generative adversarial networks (GANs) evaluated against learned discriminators (Kusner & Hernández-Lobato, 2016; Nie et al., 2019; Guo et al., 2021). While early language GANs struggled to outperform maximum likelihood (Caccia et al., 2020), those models were predominantly REINFORCE-based and targeted non-stationary discriminators rather than fixed reward models. Adjacent to LLM post-training, previous work explored reparameterized policy gradients for finetuning diffusion models (Clark et al., 2024; Prabhudesai et al., 2024) in the text-to-image generation setting. Similarly, the literature around adversarial attacks for LLMs (Wichers et al., 2024; Ebrahimi et al., 2018; Zhao et al., 2024) leverages a reward model's gradient to find adversarial examples against an LLM.

None of the works mentioned directly tackles the use of reparameterized policy gradients in the standard Transformer-based LLM post-training setup for alignment and reasoning. However, they often share some similarities with this setup, and Table 1 provides a summary of preexisting reasons in different areas that we believe are applicable as to why no significant attempt has been made to directly differentiate through a reward model to finetune LLMs. See Appendix A for additional details.

3 Background

We consider the final RL step of the standard setup for LLM post-training, in which a pretrained LM is optimized against a given reward model $r(x, y)$, where x are prompts sampled from a pre-existing dataset $\mathcal{D}$ and y are responses sampled autoregressively from a policy $\pi_\theta(y \mid x)$. The prompts and responses x, y are represented as a sequence of one hot vectors $x_1, ..., x_n \in (\Delta^{|V|-1})^n$ and $y_1, ..., y_l \in (\Delta^{|V|-1})^l$, where $|V|$ is the size of the model's vocabulary. We use the abbreviated x, y for entire sequences throughout this work. We refer to the pretrained checkpoint π_0 as the base, reference, or initial policy. Concretely, the RL objective for finetuning LLMs is

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

where y is a sequence of length l sampled autoregressively from a policy at the individual token level $\pi_\theta(y_t \mid x, y_{<t})$ until some maximum generation length L is reached or until a unique end of sentence token `<eos>` is generated. This formulation encompasses both the RLHF and the RLVR

setting depending on the choice of r; the former uses a language model with a classification head trained on preference data as its reward (Ouyang et al., 2022; Stiennon et al., 2022), while the latter defines it generally through rule-based or programmatic verifiers scoring correctness against known ground-truth answers (Lambert et al., 2025; DeepSeek-AI et al., 2025).

Score Function Estimator Since the above objective is not immediately differentiable due to its distributional dependency on the discrete sampling process $y \sim \pi_\theta$, prior work has primarily focused on using the score function estimator, also known as REINFORCE (Williams, 1992), to estimate its gradient (Ouyang et al., 2022; Shao et al., 2024; Ahmadian et al., 2024):

$$\nabla_\theta J(\theta) = \mathbb{E}_{x\sim D, y\sim\pi_\theta(\cdot|x)} \left[(r(x,y) - b)\nabla_\theta \log \pi_\theta(y \mid x)\right] \quad , \tag{2}$$

where different values for the baseline b can be used to reduce the variance of this estimator (Williams, 1992). The baseline leaves the estimator unbiased for any choice independent of y. In particular, GRPO (Shao et al., 2024) standardizes rewards within a group of sampled completions to construct a baseline, replacing $r(x,y) - b$ with $(r(x,y) - \text{mean}(r))/\text{std}(r)$, and further combines this with a PPO-style clipped surrogate objective for stable learning.

Kullback-Leibler Constraint In many cases, especially for RLHF, the reverse Kullback-Leibler (KL) divergence with respect to the base policy and the current policy is often added as a penalty to Equation 1 to avoid reward over-optimization (Gao et al., 2022). It has become standard practice to optimize directly through the per-token KL penalty instead of folding it into the reward function by considering the approximate `K3` estimator (Schulman, 2020) added to the reward loss:

$$J_\beta(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)} \left[r(x,y)\right] - \beta \texttt{K3}[\pi_\theta \| \pi_0] \quad , \quad \text{where} \tag{3}$$

$$\texttt{K3}[\pi_\theta \| \pi_0] \coloneqq \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)} \left[\sum_{t=1}^{L} \frac{\pi_0(y_t|x, y_{<t})}{\pi_\theta(y_t|x, y_{<t})} - \log \frac{\pi_0(y_t|x, y_{<t})}{\pi_\theta(y_t|x, y_{<t})} - 1\right] \quad . \tag{4}$$

Gumbel Softmax Trick The one-hot response tokens y_t can also be sampled exactly by sampling $\epsilon \sim G(0,1)$ from the standard Gumbel distribution and applying an $\arg\max$ over the noisy log probabilities (Jang et al., 2016), $y_t = \arg\max_v \log \pi_\theta(v \mid x, y_{<t}) + \epsilon_t$. This reparameterization of categorical sampling allows us to replace the discontinuous $\arg\max$ with a tempered softmax, yielding a differentiable relaxation of the one-hot token. Specifically, for a categorical distribution p, Gumbel noise ϵ, and temperature τ, we define:

$$\text{GS}(p, \epsilon; \tau) \coloneqq \text{Softmax}\big((\log p + \epsilon)/\tau\big) \quad .$$

4 Terminal Aware Gumbel Backpropagation-based Policy Optimization

Before introducing a reparameterized gradient for Equation 1, we impose two additional assumptions on the reward function. The first concerns differentiability, which is necessary to enable BPO. The second requires that the reward function and the policy share a common vocabulary, such that one-hot vector representations of the sampled responses y can be directly fed into the reward function without being decoded and re-encoded by a separate tokenizer, since this is usually non-differentiable.

Assumption 1 (Differentiability)**.** *The reward function $r(x,y)$ is fully differentiable with respect to its inputs.*

Assumption 2 (Shared vocabulary)**.** *The reward function $r(x,y)$ shares the same vocabulary as the policy $\pi_\theta(y|x)$.*

The standard transformer architecture of modern LMs satisfies the first assumption by transforming discrete tokens, represented as a probability distribution $y_t \in \Delta^{|V|-1}$, into corresponding continuous embeddings $e_t \in \mathbb{R}^d$ through the product of y_t and some learned embedding table $E \in \mathbb{R}^{|V|\times d}$. The same operations can then be applied to any soft approximation $\hat{y} \in \Delta^{|V|-1}$ of the hard one-hot tokens. In practice, the second assumption is also trivially satisfied in post-training regimes where a reward model is learned on top of the same base model as the policy, as is often the case in previous

work (Ouyang et al., 2022; Ahmadian et al., 2024). Applying the Gumbel Softmax trick naively to the objective in Equation 1 yields the following gradient:

$$\nabla_\theta^{\text{Naive-BPO}} J(\theta) = \mathbb{E}_{x\sim\mathcal{D},\epsilon\sim G(0,1)}\left[\nabla_\theta r(x,\hat{y})\right] \quad , \quad \text{where} \quad \hat{y}_t = \text{GS}\big(\pi_\theta(\cdot \mid x, \hat{y}_{<t}), \epsilon_t; \tau\big) \tag{5}$$

and the generation length l of $\hat{y}$ is still determined by the first instance of an <eos> generated by the $\arg\max$ of $\hat{y}_t$. At this point, it may be becoming apparent that the exact mechanism surrounding the terminal condition would be worth making explicit to ensure we are not missing any implicit dependencies $\hat{y}$ may have with respect to the policy parameters.

4.1 Differentiating Through the Terminal Condition

We clarify the generation process $y \sim \pi_\theta(\cdot \mid x)$ as it relates to the terminal condition. Consider the non-terminating sequence with maximum generation length L sampled from the non-terminating policy $y \sim \overline{\pi}_\theta(\cdot \mid x)$ where $\overline{\pi}_\theta$ differs from π_θ only by setting the probability of <eos> to zero and normalizing accordingly. Further, let $\mathbb{P}_\theta(l \mid x, y)$ be the probability that the first occurrence of an <eos> token in the prompt-response pair x, y is sampled at the l-th token of the response y by the policy π_θ. Then, we rewrite the objective in Equation 1 as

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D},y\sim\overline{\pi}_\theta,l\sim\mathbb{P}_\theta(\cdot|x,y)}\left[r(x,y_{<l})\right], \quad \text{where}$$

$$\mathbb{P}_\theta(l|x,y) = \begin{cases} \pi_\theta(y_l = \texttt{<eos>} \mid x, y_{<l}) \prod_{j=1}^{l-1} \pi_\theta(y_j \neq \texttt{<eos>} \mid x, y_{<j}) & \text{if } l \leq L \\ \prod_{j=1}^{l} \pi_\theta(y_j \neq \texttt{<eos>} \mid x, y_{<j}) & \text{if } l > L \end{cases} . \tag{6}$$

While it is straightforward to reparameterize the dependency with respect to $\overline{\pi}_\theta$ using the Gumbel Softmax trick, the dependency on the terminal condition is also a function of θ. Instead of trying to also reparameterize through this dependency, we find that all possible generation lengths l can be easily enumerated given a generated sequence to obtain the following policy gradient:

$$\nabla_\theta^{\text{TAG-BPO}} J(\theta) = \mathbb{E}_{x,\epsilon\sim G(0,1)}\left[\sum_{l=1}^{L} r(x,\hat{y}_{<l})\nabla_\theta \mathbb{P}_\theta(l \mid x,\hat{y}) + \mathbb{P}_\theta(l \mid x,\hat{y})\nabla_\theta r(x,\hat{y}_{<l})\right], \tag{7}$$

where $\hat{y}_t = \text{GS}\big(\overline{\pi}_\theta(\cdot \mid x, \hat{y}_{<t}), \epsilon_t; \tau\big)$. See Appendix B.1 for a full derivation of how this is an unbiased gradient of Equation 6. Following the precedent set in several prior works (Hafner et al., 2023; Ghugare et al., 2023; Ma et al., 2024; Wichers et al., 2024), we can apply a stop-gradient operator to the policy inputs to simplify the gradients without generally affecting performance. We denote this simplified gradient with a hat notation on the gradient symbol $\widehat{\nabla}_\theta^{\text{TAG-BPO}} J(\theta)$.

The rewards for all subsequences $r(x, \hat{y}_{<l})$ for all $l \leq L$ can be obtained with a single batched forward pass through the final layer of the transformer reward model given the final reward $r(x, \hat{y}_{<L})$, due to the causal nature of the transformer architecture used for modern LMs. We additionally address cases where the sub-rewards $r(x, \hat{y}_{<l})$ are not easily attainable for whatever reason by approximating this marginalization with a score function estimator in Appendix B.3. This factorization can also be seen as an instance of variance reduction by the Rao-Blackwell theorem (Blackwell, 1947), but that is not the main focus of this work.

4.2 Decoupling Generation and Gradients

The gradient update in Equation 7 can be impractical to implement, in particular because the gradients are intertwined with the generation process. This makes it difficult to leverage existing frameworks for finetuning LLMs relating to generations such as vLLM (Kwon et al., 2023) which were not designed with differentiability in mind. To address this limitation, we derive the following backpropagation-based policy gradient that decouples generation and gradients, allowing us to compute gradients from on-policy samples y agnostic to how they were generated.

$$\widehat{\nabla}_\theta^{\text{TAG-BPO}} J(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D},\ y\sim\pi_\theta(\cdot|x) \\ \epsilon\sim G(0,1|x,y,\pi_\theta)}}\left[\sum_{j=1}^{l} r(x,y_{<j})\nabla_\theta \mathbb{P}_\theta(j \mid x,y) + \mathbb{P}_\theta(j \mid x,y)\nabla_\theta r(x,\hat{y}_{<j})\right] , \tag{8}$$

where $\hat{y}_t = \text{GS}\big(\overline{\pi}_\theta(\cdot \mid x, y_{<t}), \epsilon_t; \tau\big)$, l is the length of the sampled response y, and $G(0, 1|x, y, \pi_\theta)$ is the conditional Gumbel distribution (Tucker et al., 2017; Paulus et al., 2020) such that $\epsilon \sim$

$G(0, 1|x, y, \pi_\theta)$ satisfies $\hat{y} = y$ in the limit of $\tau \to 0$. See Appendix B.4 for additional information on how to sample from the conditional Gumbel distribution, and Appendix B.2 for a complete derivation of this estimator and how it relates to Equation 7.

Importantly, updates with Equation 8 can be made using any generation process and with a single batched forward pass through the policy, similar to updates with the score function estimator in Equation 2. Since generation is often the main bottleneck to RL updates, our BPO updates can enjoy similar asymptotic memory and compute complexities as REINFORCE style updates. There is an additional backwards pass through the reward function that scales with the reward model size, but not the sequence lengths or the size of the policy. Concretely, given any batch of B prompt-response pairs (x^i, y^i) of length l^i sampled on-policy[1] and $\epsilon^i \sim G(0, 1|x^i, y^i, \pi_\theta)$, we can use any automatic differentiation library on the following loss to update the policy with TAG-BPO:

$$\mathcal{L}_\theta^{\text{TAG-BPO}} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{l^i}\mathbb{P}_\theta(j|x^i, y^i)r(x^i, \hat{y}^i_{<j}), \quad \text{where } \hat{y}^i_t = \text{GS}\big(\overline{\pi}_\theta(\cdot \mid x^i, y^i_{<t}), \epsilon^i_t; \tau\big) \ .$$

4.3 An Unbiased Kullback-Leibler Gradient

Prior work has shown that the gradient of the K3 estimator is a biased gradient of the true reverse KL penalty we seek to minimize (Shah et al., 2026; He, 2026). In particular, we focus on the bias of this estimator as it relates to the per-token reverse KL, and refer the reader to Shah et al. (2026) and He (2026) for its bias relating to the sequence level reverse KL. Formally, it can be shown that the gradient of the `K3` estimator is equivalent to the gradient of the sum of the per-token forward KLs between π_0 and π_θ instead of the reverse KLs specified by the original objective:

$$\nabla_\theta \texttt{K3}[\pi_\theta||\pi_0] = \nabla_\theta D_{KL}[\pi_0||\pi_\theta] \ \neq \nabla_\theta D_{KL}[\pi_\theta||\pi_0] \ .$$

Concretely, this bias ignores the distributional dependency of the reverse KL with respect to the policy parameters θ, so that increases in the probability of tokens that are unlikely under the reference policy are under-penalized. We argue that the effect of this bias can be negligible for policy gradients using the score function estimator, because unlikely tokens must be sampled to be updated for these methods, which occur rarely by definition. Conversely, a reparameterized policy gradient has full coverage over a token's probability distribution and can fully leverage the bias from the `K3` estimator to disproportionately update tokens in the tail distribution of the vocabulary.

To address this bias, we suggest using the per-token analytical KL divergence suggested in Tang & Munos (2025) for TAG-BPO:

$$\nabla_\theta \texttt{KFull} := \mathbb{E}_{x\sim D, y\sim\pi_\theta(\cdot|x)}\left[\nabla_\theta \sum_{t=1}^{L}\sum_{v\in\mathcal{V}}\pi_\theta(v \mid x, y_{<t})\log\frac{\pi_\theta(v \mid x, y_{<t})}{\pi_0(v \mid x, y_{<t})}\right] \ , \tag{9}$$

which explicitly enumerates the full vocabulary $\mathcal{V}$ of the policy at every position. See Appendix B.5 for a derivation of how ∇_θ`KFull` is an unbiased estimator of the sum of the per-token reverse KL gradients, but just like `K3`, it is not an unbiased gradient of the sequence level reverse KL.

4.4 Long Horizon Gradient Stability

Backpropagation-based policy optimization is known to have unfavorable gradient properties as the horizon of an environment grows, which would make it unsuitable for finetuning LLMs where completions easily exceed dozens, hundreds, or even thousands of tokens. In particular, gradient explosions (Parmas et al., 2018) described as an exponential dependency of the gradient norm with respect to a maximum generation length L limit the use of BPO in traditional settings. However, notice that the reward gradient $\nabla_\theta r(x, \hat{y})$ in Equation 8 is not auto-regressive in structure, and instead resembles an Action-conditioned World model (AWM) as described in Ma et al. (2024); in language generation, the reward function is a direct mapping from a sequence of actions to a reward. Formally, we extend the result relating to the gradient norms of AWMs from Ma et al. (2024) for the discrete language setting and bound the gradient norm of the reward gradient term.

[1] Similarly to PPO and GRPO, off-policy samples can also be used by adding an importance sampling ratio.

Method	IMDb		TL;DR		GSM8K	
	Train ($\beta = 0$)	Eval	Train ($\beta = 0$)	Eval	Train ($\beta = 0$)	Eval
TAG-BPO	**10.31 ± 0.26**	–	**24.27 ± 1.22**	**0.84 ± 0.04**	**0.927 ± 0.517**	0.343 ± 0.100
GRPO	**10.00 ± 0.29**	–	19.92 ± 0.26	**0.90 ± 0.03**	**1.280 ± 0.172**	**0.510 ± 0.006**
REINFORCE	8.23 ± 0.10	–	11.06 ± 0.38	0.44 ± 0.10	0.888 ± 0.131	**0.509 ± 0.007**
Base	0.30	–	0.84	0.22	−10.26	0.117

Table 2: Mean (± std) training and evaluation rewards across IMDb, TL;DR, and GSM8K. Training results correspond to the unconstrained setting ($\beta = 0$), while evaluation results are obtained using the best β values for TL;DR. **Bold** indicates the best result among the methods for each column accounting for uncertainty.

Proposition 1. *Let π_θ be an L_π-Lipschitz function with respect to θ, let M be the maximum completion length, and let $\tau > 0$ be the Gumbel temperature. Assuming ϵ is independent of θ, the gradient of the reward term in Equation 8 is bounded as*

$$\|\nabla_\theta r(x, \hat{y})\| \leq \frac{L_\pi}{\tau} \sum_{t=1}^{M} \left\| \frac{\partial r(x, \hat{y})}{\partial \hat{y}_t} \right\| .$$

5 Experiments and Discussion

Experiments are performed on **IMDb** positive sentiment generation (Maas et al., 2011), **TL;DR** text summarization (Huang et al., 2024; Völske et al., 2017), and **GSM8K** math reasoning (Lambert et al., 2025; Cobbe et al., 2021) to cover tasks both in RLHF and RLVR. For all settings, we compare our method with the GRPO (Shao et al., 2024) implementation from TRL (von Werra et al., 2020) and a vanilla REINFORCE estimator which removes the baseline and clipping mechanism from the GRPO implementation to get a clearer comparison between raw gradient estimators. All experiments report the average performance over 5 independently run seeds. We give a brief overview of each task, and leave additional details in Appendix C.

IMDb: The task consists of prompt prefixes 2-8 tokens long from the IMDb dataset (Maas et al., 2011), and the goal is to generate completions with positive sentiment from these prefixes. A 124 million parameter `GPT2-small` model is used as the base model, and we train an SFT checkpoint and reward model on top of this model using the original labeled dataset. Similar to Rafailov et al. (2024), we sweep several values of β to plot the KL-vs-reward in Figure 1 to evaluate each method.

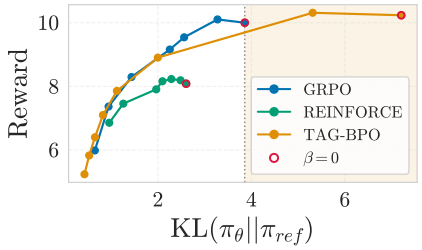

Figure 1: KL–reward trade-offs on IMDb for a 124M GPT2-small policy, obtained by sweeping the KL coefficient β.

TL;DR: We follow the setup used in Huang et al. (2024) and borrow the policy and reward models used in the experiments in Noukhovitch et al. (2025), where smaller reward models were trained on a dataset of preferences relabeled by a larger oracle reward model, considered the ground truth or gold reward model (Gao et al., 2022). In this case, we use reward and policy models from the Pythia family (Biderman et al., 2023) of size 2.8B against a gold 6.9B parameter reward model. Evaluation is done by scoring completions from a trained model against human completions with the gold reward model, and reporting how often the trained model's completions outscore the human completions, otherwise known as a win-rate. We report the best evaluation win-rate across several values of β to mitigate reward overoptimization.

GSM8K: For reasoning, we use a Qwen3-0.6B-Base (Yang et al., 2025) model as the initial policy, and follow the one-shot prompt setup used in Ackermann et al. (2026) popularized by Hugging Face (2025). We use a more lenient correctness function we defined ourselves that is less strict on exact formatting than the correctness function provided in Ackermann et al. (2026) (Appendix C.3.1). Since neither of the correctness rewards are differentiable, we consider a Skywork-Reward-V2-Qwen3-0.6B (Liu et al., 2026) as a differentiable alternative for training, and use our more lenient correctness function for evaluation. Following the precedent set by prior work (Yu et al., 2025; Ackermann et al., 2026), we do not use a KL penalty for this task.

Method	TL;DR 410M		TL;DR 1B		TL;DR 2.8B	
	Train ($\beta = 0$)	Eval	Train ($\beta = 0$)	Eval	Train ($\beta = 0$)	Eval
TAG-BPO	**13.61 ± 1.50**	0.18 ± 0.08	**13.07 ± 0.89**	0.51 ± 0.01	**24.27 ± 1.22**	**0.84 ± 0.04**
GRPO	10.46 ± 0.69	**0.47 ± 0.03**	10.52 ±1.01	**0.74 ± 0.05**	19.92 ± 0.26	**0.90 ± 0.03**
Base	0.25	0.08	0.89	0.13	0.84	0.22

Table 3: **TAG-BPO consistently achieves higher training rewards than GRPO, but its evaluation performance degrades more sharply as model size decreases.** Mean (± std) training and evaluation results on TL;DR for 410M, 1B, and 2.8B policies and reward models. Training results correspond to the unconstrained setting ($\beta = 0$), while evaluation results report the best win-rate across the tested values of β. **Bold** indicates the best result among the trained methods for each column accounting for uncertainty.

The training and evaluation results are presented in Table 2 for each task and different method. Training rewards are reported for the unconstrained setting ($\beta = 0$), and evaluation scores are reported for the best β values for TL;DR. The goal of these experiments is not to show that a reparameterized policy gradient is necessarily better than current state-of-the-art methods such as GRPO, but that it is a viable alternative for LLM post-training despite the lack of interest from the literature. Moreover, for similar hyperparameters, TAG-BPO has a comparable compute and memory cost to a score function estimator; training is 1.2 to 1.3 times slower than GRPO and REINFORCE, and peak GPU memory usage is 1.1 to 1.35 times higher than the score function baselines across tasks with varying completion lengths and model sizes (Appendix C.4).

5.1 Ablations

Model Size. To determine the impact that model size has on TAG-BPO, we focus on TL;DR which is often studied for scaling properties (Gao et al., 2022). We decrease both the policy and reward model sizes to 410M and 1B parameters, since we are restricted by the size of our gold reward model. Just as before, we report both the best training reward achieved in the unconstrained setting, and the best evaluation win-rates across different values of β for TAG-BPO and GRPO in Table 3. The results reveal a notable disadvantage of reparameterized gradients with respect to reward hacking. Even though our method consistently outperforms GRPO as a reward maximizer for $\beta = 0$, its evaluation win-rates are increasingly worse as the model strays further away from the gold model.

Terminal Aware. We empirically validate the necessity of the terminal aware policy gradient by comparing training curves of TAG-BPO vs the Naive-BPO estimator introduced in Equation 5 in every task without a KL constraint, since the KL constraint can implicitly control the completion lengths. We use the smallest 410M parameter Pythia models for TL;DR for this ablation, but keep the same setup for IMDb and GSM8K. The training rewards shown in Figure 2 show a clear gap in performance in the absence of a terminal aware policy gradient. For TL;DR and IMDb, the completions quickly degenerate towards extreme lengths, but the difference is more subtle for GSM8K. The Skywork reward model used to train the reasoning task does not share the same <eos> token as the base Qwen3-0.6B-Base policy, making it more difficult to confound credit on incorrect terminal conditions. For a more intuitive experiment on the effect of differentiating through the terminal condition, see the toy example in Appendix D.

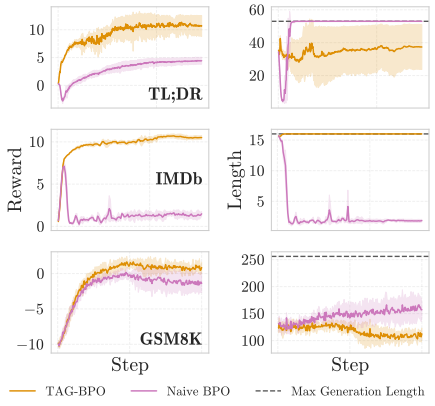

Figure 2: Training reward curves and completion lengths comparing TAG-BPO and Naive-BPO without a KL constraint across IMDb, TL;DR, and GSM8K.

Full KL Divergence. We perform an ablation on the effects of the `KFull` estimator for TAG-BPO on different model sizes for the TL;DR task, and whether GRPO could also benefit from the same KL estimator. The results in Figure 3 show that the reparameterized policy gradient traces a strictly better KL-vs-reward frontier when trained with `KFull`, while the performance of GRPO remains similar in either case. As discussed in Section 4.3, the gradient bias of the `K3` loss towards the forward KL divergence is more meaningful under TAG-BPO than GRPO due to its full coverage of updates over rarely sampled tokens. We perform the same ablation for IMDb in Appendix E.

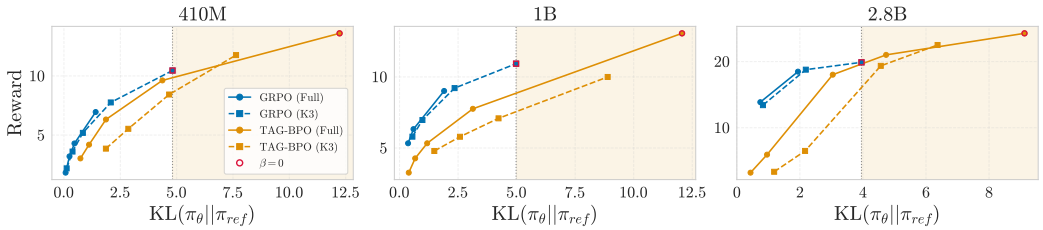

Figure 3: **`KFull` yields a better KL–reward frontier than K3 for TAG-BPO, while the choice of KL estimator has little effect on GRPO.** KL–reward trade-offs on TL;DR comparing the `K3` and `KFull` estimators for both TAG-BPO and GRPO across different model sizes: 410M, 1B, 2.8B.

5.2 Escaping the Base Policy

The KL-vs-reward tradeoff plots in Figure 1 and Figure 3 show an interesting trend where TAG-BPO is more likely to update the policy towards higher KL regions with potentially higher training rewards that GRPO is unable to reach even when $\beta = 0$. We find this behavior to be somewhat consistent in GSM8K, but the effect to be less pronounced (Appendix F).

To further demonstrate the potential benefits of this phenomenon, we run a simple experiment on a variant of the GSM8K task in which we replace the reward function with a custom differentiable reward that counts occurrences of a predetermined set of target tokens anywhere in the completion, regardless of context. The optimal, fully degenerate policy therefore fills its entire completion with target tokens alone. We consider two target sets that differ sharply in how likely they already are under the base policy: variations of the out-of-context word "apple," which are very rarely generated by the reference policy ($\sim 0.02\%$), and any token that decodes to a digit ($\sim 9.7\%$). Figure 4 shows the resulting training curves for TAG-BPO and GRPO on both target sets. As advantages are zero in almost every completion for the former, GRPO completely fails to update its policy. Conversely, TAG-BPO quickly solves the task without needing to sample the appropriate token to start making progress. Even when the target tokens are sampled in the latter case, GRPO struggles to find the optimal solution due to its tendency to stick to the base policy.

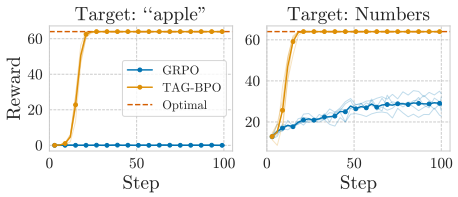

Figure 4: Training curves on the toy target-token task with GSM8K prompts and a Qwen3-0.6B-Base policy, for a target set with extremely rare tokens ("apple") and a more likely set (numbers).

6 Conclusion

We introduced TAG-BPO, a reparameterized policy gradient for finetuning LLMs with RL. TAG-BPO removes two main obstacles that have kept this class of estimators out of the language modeling literature: (1) it differentiates through the terminal stopping condition explicitly, and (2) its gradient is not dependent on the generation process, making it an easy substitute in existing post-training frameworks with little additional cost. We also show that, unlike traditional RL environments, the unique structure of LLM post-training naturally avoids the long-horizon gradient instability historically attributed to backpropagation-based policy optimization. Empirically, TAG-BPO attains higher or similar training rewards to GRPO across all environments we study, but its gains do not always transfer directly to held-out or gold rewards: when the training reward model is limited, TAG-BPO is more susceptible to reward over-optimization, while its evaluation performance approaches that of GRPO as the reward model is scaled. We find that these effects are closely tied to TAG-BPO's greater willingness to move beyond the base policy's distribution: this allows TAG-BPO to discover higher-reward completions in regions that score function methods such as GRPO rarely explore, but also exposes it to reward hacking.

To the best of our knowledge, our work is the first to explore and fully deliver a practical estimator for reparameterized policy gradients in the LLM post-training regime. Despite the limited scope of our experiments, we believe TAG-BPO lays an important foundation towards better understanding the possible advantages and disadvantages of backpropagation-based policy optimization for language modeling.

AI USE STATEMENT

In this work, we have not used generative AI tools to assist in the writing of proofs, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine a hypothesis, design or provide feedback on research methodology or experiments, implement methods, assist with translation, interpret results. and generating synthetic data sets, cleaning and reformatting dataset, supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to discover research topics or identify gaps, summarize existing literature, double-check mathematical claims and proofs, and create or edit software code. We have reviewed all AI-assisted work. All suggested research topics and references were manually read, and all LLM-generated code has been verified and tested by two authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Johannes Ackermann, Michael Noukhovitch, Takashi Ishida, and Masashi Sugiyama. Gradient regularization mitigates reward hacking in reinforcement learning from human feedback and verifiable rewards, 2026. URL https://arxiv.org/abs/2602.18037.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017. URL https://arxiv.org/abs/1701.07875.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback, April 2022.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling, 2023. URL https://arxiv.org/abs/2304.01373.

David Blackwell. Conditional expectation and unbiased sequential estimation. *The Annals of Mathematical Statistics*, pp. 105–110, 1947.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. Language GANs Falling Short, February 2020.

Pengxiang Cai, Tianchen Fang, Xiaohan Li, Qingyuan Zeng, Guocong Li, and Jintai Chen. Curriculum reinforcement learning can incentivize reasoning capacity in llms beyond the base model, 2026. URL https://arxiv.org/abs/2606.22317.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards, 2024. URL https://arxiv.org/abs/2309.17400.

Ignasi Clavera, Yao Fu, and Pieter Abbeel. Model-augmented actor-critic: Backpropagating through paths. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Skln2A4YDB.

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. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The Entropy Mechanism of Reinforcement Learning for Reasoning Language Models, May 2025.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao,

Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *Nature*, 645(8081):633–638, September 2025. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-025-09422-z.

Marc Peter Deisenroth and Carl Edward Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. 2011.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification, 2018. URL https://arxiv.org/abs/1712.06751.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization, 2022. URL https://arxiv.org/abs/2210.10760.

Ignat Georgiev, Krishnan Srinivasan, Jie Xu, Eric Heiden, and Animesh Garg. Adaptive Horizon Actor-Critic for Policy Learning in Contact-Rich Differentiable Simulation, June 2024.

Raj Ghugare, Homanga Bharadhwaj, Benjamin Eysenbach, Sergey Levine, and Ruslan Salakhutdinov. Simplifying model-based rl: Learning representations, latent-space models, and policies with one objective, 2023.

Audrūnas Gruslys, Remi Munos, Ivo Danihelka, Marc Lanctot, and Alex Graves. Memory-efficient backpropagation through time, 2016. URL https://arxiv.org/abs/1606.03401.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans, 2017. URL https://arxiv.org/abs/1704.00028.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. Gradient-based Adversarial Attacks against Text Transformers, April 2021.

Danijar Hafner, Timothy P. Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=0oabwyZbOu.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models, 2022. URL https://arxiv.org/abs/2010.02193.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2023.

Muyu He. Computing the gradient of the KL penalty. Blog post, June 2026. URL https://riddlehe.github.io/blog/kl-gradient-estimators.html. Accessed: 2026-09-21.

Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. *Advances in neural information processing systems*, 28, 2015.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. Toward Controlled Generation of Text, September 2018.

Shengyi Huang, Michael Noukhovitch, Arian Hosseini, Kashif Rasul, Weixun Wang, and Lewis Tunstall. The n+ implementation details of rlhf with ppo: A case study on tl;dr summarization, 2024. URL https://arxiv.org/abs/2403.17031.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1. https://github.com/huggingface/open-r1, 2025.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Xiaojun Jia, Tianyu Pang, Chao Du, Yihao Huang, Jindong Gu, Yang Liu, Xiaochun Cao, and Min Lin. Improved techniques for optimization-based jailbreaking on large language models, 2024. URL https://arxiv.org/abs/2405.21018.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms, 2025. URL https://arxiv.org/abs/2410.01679.

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the Effects of RLHF on LLM Generalisation and Diversity, February 2024.

Matt J. Kusner and José Miguel Hernández-Lobato. GANS for Sequences of Discrete Elements with the Gumbel-softmax Distribution, November 2016.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing Frontiers in Open Language Model Post-Training, April 2025.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, Yang Liu, and Yahui Zhou. Skywork-reward-v2: Scaling preference data curation via human-ai synergy, 2026. URL https://arxiv.org/abs/2507.01352.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models, 2025a. URL https://arxiv.org/abs/2505.24864.

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, October 2025b.

Michel Ma, Tianwei Ni, Clement Gehring, Pierluca D'Oro, and Pierre-Luc Bacon. Do transformer world models give better policy gradients? *arXiv preprint arXiv:2402.05290*, 2024.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Chris J. Maddison, Daniel Tarlow, and Tom Minka. A* sampling, 2015. URL https://arxiv.org/abs/1411.0030.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Luke Metz, C. Daniel Freeman, Samuel S. Schoenholz, and Tal Kachman. Gradients are Not All You Need, January 2022.

Weili Nie, Nina Narodytska, and Ankit B Patel. RELGAN: RELATIONAL GENERATIVE ADVERSARIAL NETWORKS FOR TEXT GENERATION. 2019.

Michael Noukhovitch, Shengyi Huang, Sophie Xhonneux, Arian Hosseini, Rishabh Agarwal, and Aaron Courville. Asynchronous rlhf: Faster and more efficient off-policy rl for language models, 2025. URL https://arxiv.org/abs/2410.18252.

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

Paavo Parmas, Carl Edward Rasmussen, Jan Peters, and Kenji Doya. Pipps: Flexible model-based policy search robust to the curse of chaos. In *International Conference on Machine Learning*, pp. 4065–4074. PMLR, 2018.

Paavo Parmas, Takuma Seno, and Yuma Aoki. Model-based Reinforcement Learning with Scalable Composite Policy Gradient Estimators. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 27346–27377. PMLR, July 2023.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks, 2013. URL https://arxiv.org/abs/1211.5063.

Max B Paulus, Chris J Maddison, and Andreas Krause. Rao-blackwellizing the straight-through gumbel-softmax gradient estimator. *arXiv preprint arXiv:2010.04838*, 2020.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of machine learning and systems*, 5:606–624, 2023.

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation, 2024. URL https://arxiv.org/abs/2310.03739.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model, July 2024.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021. URL https://arxiv.org/abs/2102.12092.

Schulman. Approximating KL Divergence. http://joschu.net/blog/kl-approx.html, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, August 2017.

Vedant Shah, Johan Obando-Ceron, Vineet Jain, Brian Bartoldson, Bhavya Kailkhura, Sarthak Mittal, Glen Berseth, Pablo Samuel Castro, Yoshua Bengio, Esmeralda S. Whitammer, Moksh Jain, Siddarth Venkatraman, and Aaron Courville. A comedy of estimators: On kl regularization in rl training of llms, 2026. URL https://arxiv.org/abs/2512.21852.

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.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback, February 2022.

H. J. Terry Suh, Max Simchowitz, Kaiqing Zhang, and Russ Tedrake. Do Differentiable Simulators Give Better Policy Gradients?, August 2022.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation.

Yunhao Tang and Rémi Munos. On a few pitfalls in kl divergence gradient estimation for rl, 2025. URL https://arxiv.org/abs/2506.09477.

George Tucker, Andriy Mnih, Chris J. Maddison, Dieterich Lawson, and Jascha Sohl-Dickstein. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models, 2017. URL https://arxiv.org/abs/1703.07370.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. Tl; dr: Mining reddit to learn automatic summarization. In *Proceedings of the workshop on new frontiers in summarization*, pp. 59–63, 2017.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

Nevan Wichers, Carson Denison, and Ahmad Beirami. Gradient-based language model red teaming, 2024. URL https://arxiv.org/abs/2401.16656.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

Fang Wu, Weihao Xuan, Ximing Lu, Mingjie Liu, Yi Dong, Zaid Harchaoui, and Yejin Choi. The invisible leash: Why rlvr may or may not escape its origin, 2026. URL https://arxiv.org/abs/2507.14843.

Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated Policy Learning with Parallel Differentiable Simulation, April 2022.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model?, November 2025.

Oussama Zekri and Nicolas Boullé. Fine-tuning discrete diffusion models with policy gradient methods, 2025. URL https://arxiv.org/abs/2502.01384.

Yiran Zhao, Wenyue Zheng, Tianle Cai, Xuan Long Do, Kenji Kawaguchi, Anirudh Goyal, and Michael Shieh. Accelerating greedy coordinate gradient and general prompt optimization via probe sampling, 2024. URL https://arxiv.org/abs/2403.01251.

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, January 2020.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023. URL https://arxiv.org/abs/2307.15043.

A Extended Related Works for Reparameterized Gradients

To the best of our knowledge, we are not aware of any related works that specifically use a reward model's gradient information to do LLM post-training in the RLHF or RLVR literature. Since this seems like a surprising gap, we list and cite a non-exhaustive list of reasons why we think it has not been tried yet.

Cost Concerns It is well known that the gradient through a sequence model incurs a non-negligible cost in accumulating gradients either in forward mode or reverse mode automatic differentiation (Williams & Zipser, 1989; Werbos, 1990). However, the actual cost in memory, in the case of backpropagation through time, can generally be mitigated with gradient checkpointing (Chen et al., 2016; Gruslys et al., 2016). Instead, the primary bottleneck for RLHF and RLVR is the cost of inference from auto-regressively generating completions rather than being able to parallelize generations across time steps (Kwon et al., 2023; Pope et al., 2023). Requiring gradient information to be stored during generation adds a layer of complexity for efficient generation frameworks, and limits the use of reparameterized gradients in this setting.

Unstable Long-horizon Gradients Due to the fact that current frontier language models are fundamentally auto-regressive in nature, they inherit the long-horizon gradient problems well documented in sequence modeling (Pascanu et al., 2013) and model-based RL (Parmas et al., 2018; Suh et al., 2022). These difficulties can often be traced back to an exponential dependency of the model Jacobians with respect to the horizon of a problem. Given the current generation lengths of LLMs, it is natural to be cautious of these methods for LLM post-training.

Relaxation Bias In order to make a discrete environment differentiable, it is necessary to bias the gradient estimate using some form of soft approximation (Jang et al., 2016). While this bias has seen success in different machine learning fields (Ramesh et al., 2021; Maddison et al., 2016), attempts to use it in practice for Reinforcement Learning has generally been unsuccessful (Hafner et al., 2021). Instead, a score function estimator is always unbiased, and is generally favored in discrete RL environments.

First-order Approximation of a Reward Model's Curvature At its core, a reparameterized policy gradient is a first-order gradient (Suh et al., 2022) as opposed to the zero-th order estimate of REINFORCE. Depending on the reward model's landscape with respect to its inputs, a first-order gradient estimate can run into several optimization problems. This phenomenon has been well documented in the GANs literature (Arjovsky et al., 2017; Gulrajani et al., 2017), where the discriminator and generator play a similar role to the reward model and policy of the LLM finetuning setup. Moreover, works in adversarial attacks against LLMs that do directly use the gradient of a reward model ($\nabla_{x,y} r(x,y)$) to find vulnerabilities in LLMs have already noted how unreliable that gradient is for finding global maximums (Ebrahimi et al., 2018; Wichers et al., 2024; Jia et al., 2024). While these works do not study this shortcoming in the context of a policy gradient ($\nabla_\theta r(x,y)$), our policy gradient also relies on the Jacobian $\nabla_{x,y} r(x,y)$ to make updates to the policy.

B Proofs and Derivations

B.1 Derivation of Equation 7

First, we show that the reparameterized gradient of Equation 6 is equivalent to the TAG-BPO gradient in Equation 7 without the stop gradient approximation in the limit that $\tau \to 0$:

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \mathbb{E}_{x\sim\mathcal{D}, y\sim\overline{\pi}_\theta, l\sim\mathbb{P}(\cdot|x,y,\pi_\theta)}\left[r(x, y_{<l})\right] \\
&= \mathbb{E}_{x\sim\mathcal{D},\epsilon\sim G(0,1)}\left[\sum_{l=1}^{L} r(x,\hat{y}_{<l})\nabla_\theta \mathbb{P}_\theta(l|x,\hat{y}) + \mathbb{P}_\theta(l|x,\hat{y})\nabla_\theta r(x,\hat{y}_{<l})\right], \text{where} \\
\hat{y}_t &= \mathrm{GS}\big(\overline{\pi}_\theta(\cdot \mid x, \hat{y}_{<t}), \epsilon_t; \tau\big), \\
\mathbb{P}_\theta(l|x,y) &= \begin{cases} \pi_\theta(y_l = \texttt{<eos>}|x, y_{<l}) \prod_{j=1}^{l-1} \pi_\theta(y_j \neq \texttt{<eos>}|x, y_{<j}) & \text{if } l \leq L \\ \prod_{j=1}^{l} \pi_\theta(y_j \neq \texttt{<eos>}|x, y_{<j}) & \text{if } l > L \end{cases}.
\end{aligned}
$$

Proof. Begin by rewriting the original objective in Equation 6 by factorizing out the stopping condition l:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\overline{\pi}_\theta}\left[\sum_{l=1}^{L} r(x, y_{<l})\, \mathbb{P}_\theta(l|x, y)\right] .$$

Then, using the Gumbel Softmax relaxation $\hat{y}_t = \mathrm{GS}\big(\overline{\pi}_\theta(\cdot \mid x, \hat{y}_{<t}), \epsilon_t; \tau\big)$, which is equivalent to y_t when $\tau \to 0$ (Jang et al., 2016), the objective can be reparameterized as:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D}, \epsilon\sim G(0,1)}\left[\sum_{l=1}^{L} r(x, \hat{y}_{<l})\, \mathbb{P}_\theta(l|x, \hat{y})\right] .$$

Finally, applying the gradient operator and the chain rule:

$$\nabla_\theta J(\theta) = \mathbb{E}_{x\sim\mathcal{D}, \epsilon\sim G(0,1)}\left[\sum_{l=1}^{L} r(x, \hat{y}_{<l})\nabla_\theta \mathbb{P}_\theta(l|x, \hat{y}) + \mathbb{P}_\theta(l|x, \hat{y})\nabla_\theta r(x, \hat{y}_{<l})\right].$$

□

B.2 Derivation of Equation 8

For the derivation of the decoupled TAG-BPO gradient, we begin with the TAG-BPO gradient of Equation 7, that is the reparameterized policy gradient under the stop-gradient assumption on policy inputs, and prove its equivalence to the decoupled gradient:

$$\begin{aligned}\widehat{\nabla}_\theta^{\text{TAG-BPO}} J(\theta) &= \mathbb{E}_{x\sim\mathcal{D}, \epsilon\sim G(0,1)}\left[\sum_{l=1}^{L} r(x, \hat{y}_{<l})\nabla_\theta \mathbb{P}_\theta(l|x, \hat{y}) + \mathbb{P}_\theta(l|x, \hat{y})\nabla_\theta r(x, \hat{y}_{<l})\right] \\ &= \mathbb{E}_{\substack{x\sim\mathcal{D},\ y\sim\overline{\pi}_\theta \\ \epsilon\sim G(0,1|x,y,\overline{\pi}_\theta)}}\left[\sum_{l=1}^{L} r(x, y_{<l})\nabla_\theta \mathbb{P}_\theta(l|x, y) + \mathbb{P}_\theta(l|x, y)\nabla_\theta r(x, \hat{y}_{<l})\right] , \quad (10)\end{aligned}$$

where $\hat{y}_t = \mathrm{GS}\big(\overline{\pi}_\theta(\cdot \mid x, \mathbf{sg}[\hat{y}_{<t}]), \epsilon_t; \tau\big)$ in the first line, and $\hat{y}_t = \mathrm{GS}\big(\overline{\pi}_\theta(\cdot \mid x, y_{<t}), \epsilon_t; \tau\big)$ in the second line, as in Equation 8.

The derivation is similar to the derivation found in the Appendix of Heess et al. (2015), where they similarly derive a reparameterized policy gradient conditioned on real sampled trajectories.

Proof. Let $\epsilon \sim G(0, 1)$ be the Gumbel noise. Let $p(y|x, \epsilon)$ be the probability of sampling a rollout y given the sequence of independently sampled Gumbel noise ϵ.

First, we use Bayes' rule to derive the following equivalence:

$$\begin{aligned} p(y, \epsilon, x) &= p(\epsilon, y, x) \\ p(y|x, \epsilon) &= \frac{p(\epsilon|x, y)p(y|x)}{p(\epsilon)} \ \text{ because } \epsilon \perp x, \end{aligned}$$

where $p(\epsilon|x, y) = G(0, 1 \mid x, y, \overline{\pi}_\theta)$ is the conditional Gumbel distribution of Appendix B.4.

Then, we expand the reparameterized gradient of Equation 7, writing

$$g_\theta(x, y, \epsilon) := \sum_{l=1}^{L} r(x, y_{<l})\nabla_\theta \mathbb{P}_\theta(l|x, y) + \mathbb{P}_\theta(l|x, y)\nabla_\theta r(x, \hat{y}_{<l}),$$

where y has been substituted for the relaxed sequence everywhere it appears as a discrete argument or conditioning context, the relaxed $\hat{y}_t = \mathrm{GS}\big(\overline{\pi}_\theta(\cdot \mid x, y_{<t}), \epsilon_t; \tau\big)$ being retained only in the reward

term, which is the sole place a gradient flows into the sequence:

$$\begin{aligned}
\widehat{\nabla}_\theta^{\text{TAG-BPO}} J(\theta) &= \mathbb{E}_{x\sim\mathcal{D},\epsilon\sim G(0,1)}\mathbb{E}_{y\sim p(y|x,\epsilon)} g_\theta(x,y,\epsilon) \\
&= \sum_y \sum_x \int_\epsilon g_\theta(x,y,\epsilon)\, p(\epsilon)p(x)p(y|x,\epsilon)\, d\epsilon \\
&= \sum_y \sum_x \int_\epsilon g_\theta(x,y,\epsilon)\, \cancel{p(\epsilon)}p(x)\frac{p(\epsilon|x,y)p(y|x)}{\cancel{p(\epsilon)}}\, d\epsilon \\
&= \sum_y \sum_x \int_\epsilon g_\theta(x,y,\epsilon)\, p(x)p(\epsilon|x,y)p(y|x)\, d\epsilon \\
&= \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\overline{\pi}_\theta(y|x)}\mathbb{E}_{\epsilon\sim G(0,1|x,y,\overline{\pi}_\theta)} g_\theta(x,y,\epsilon) \\
&= \mathbb{E}_{\substack{x\sim\mathcal{D},\, y\sim\overline{\pi}_\theta \\ \epsilon\sim G(0,1|x,y,\overline{\pi}_\theta)}} \left[\sum_{l=1}^{L} r(x,y_{<l})\nabla_\theta \mathbb{P}_\theta(l|x,y) + \mathbb{P}_\theta(l|x,y)\nabla_\theta r(x,\hat{y}_{<l})\right].
\end{aligned}$$

□

B.3 Score Function Approximation of the Terminal Marginalization

In cases where the rewards $r(x, y_{<l})$ are not easily attainable given $r(x, y_{1:L})$, the contribution of the terminal condition in the TAG-BPO policy gradient can instead be estimated using a score function estimator:

$$\begin{aligned}
\widehat{\nabla}_\theta^{\text{TAG-BPO}} J(\theta) &= \mathbb{E}_{\substack{x\sim\mathcal{D},\, y\sim\pi_\theta \\ \epsilon\sim G(0,1|x,y,\pi_\theta)}} \left[\sum_{j=1}^{l} r(x,y_{<j})\nabla_\theta \mathbb{P}_\theta(j \mid x,y) + \mathbb{P}_\theta(j \mid x,y)\nabla_\theta r(x,\hat{y}_{<j})\right] \\
&= \mathbb{E}_{\substack{x\sim\mathcal{D},\, y\sim\pi_\theta \\ \epsilon\sim G(0,1|x,y,\pi_\theta)}} \left[r(x,y)\nabla_\theta \log \mathbb{P}_\theta(l \mid x,y) + \nabla_\theta r(x,\hat{y})\right],
\end{aligned}$$

where l here denotes the sampled length of y.

B.4 Sampling from the Conditional Gumbel Distribution

Equation 8 requires noise drawn from the conditional Gumbel distribution $\epsilon \sim G(0, 1 \mid x, y, \pi_\theta)$, that is, standard Gumbel noise conditioned on the event that the Gumbel-max trick applied to the policy's logits reproduces the observed response y. This can be sampled exactly and in closed form as was shown in Maddison et al. (2015) and Paulus et al. (2020). We also ignore the dependency of the conditional Gumbel distribution with θ in our gradients, following the notes found in the implementation of Paulus et al. (2020).

For any position t, write the policy's log-probabilities as $z_t := \log \overline{\pi}_\theta(\cdot \mid x, y_{<t}) \in \mathbb{R}^{|V|}$ for the policy's log-probabilities at that position. Given the realized token $\arg\max y_t = i$ and i.i.d. $E_j \sim \text{Exponential}(1)$, the perturbed logits admit the equivalence in distribution

$$z_{t,j} + \epsilon_{t,j} \mid i \;\overset{d}{=}\; \begin{cases} -\log E_i & \text{if } j = i, \\ -\log\left(\dfrac{E_j}{\overline{\pi}_\theta(y = j \mid x, y_{<t})} + E_i\right) & \text{otherwise,} \end{cases} \tag{11}$$

from which ϵ_t is recovered by subtracting θ_t.

B.5 Unbiasedness of the Full KL Gradient used in Equation 9

The suggested $\nabla_\theta \texttt{KFull}$ is:

$$\nabla_\theta \texttt{KFull} := \mathbb{E}_{x\sim D, y\sim\pi_\theta(\cdot|x)} \left[\nabla_\theta \sum_{t=1}^{L}\sum_{v\in\mathcal{V}} \pi_\theta(v \mid x, y_{<t}) \log \frac{\pi_\theta(v \mid x, y_{<t})}{\pi_0(v \mid x, y_{<t})}\right],$$

and we would like to show that it is equivalent to the sum of the per-token gradient of the reverse KLs:

$$\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[\sum_{t=1}^{L}\nabla_\theta D_{KL}\left[\pi_\theta(\cdot\mid x,y_{<t})\|\pi_0(\cdot\mid x,y_{<t})\right]\right]$$

Proof.

$$\begin{aligned}
&\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[\sum_{t=1}^{L}\nabla_\theta D_{KL}\left[\pi_\theta(\cdot\mid x,y_{<t})\|\pi_0(\cdot\mid x,y_{<t})\right]\right]\\
&\quad=\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[\sum_{t=1}^{L}\nabla_\theta\mathbb{E}_{v\sim\pi_\theta(\cdot|x,y_{<t})}\left[\log\frac{\pi_\theta(v|x,y_{<t})}{\pi_0(v|x,y_{<t})}\right]\right]\\
&\quad=\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[\sum_{t=1}^{L}\nabla_\theta\sum_{v\in\mathcal{V}}\pi_\theta(v\mid x,y_{<t})\log\frac{\pi_\theta(v|x,y_{<t})}{\pi_0(v|x,y_{<t})}\right]\\
&\quad=\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(\cdot|x)}\left[\nabla_\theta\sum_{t=1}^{L}\sum_{v\in\mathcal{V}}\pi_\theta(v\mid x,y_{<t})\log\frac{\pi_\theta(v|x,y_{<t})}{\pi_0(v|x,y_{<t})}\right]
\end{aligned}$$

□

B.6 Proof of Proposition 1

First, we formalize the assumption made on an L_π-Lipschitz policy and restate the proposition:

Assumption 3 (Lipschitz policy)**.** *Let $z_\theta(x,y_{<t})\in\mathbb{R}^{|V|}$ denote the policy's logits at position t. There exists $L_\pi<\infty$ such that $\|\partial z_\theta(x,y_{<t})/\partial\theta\|\le L_\pi$ for all θ and all contexts $(x,y_{<t})$.*

Proposition 1. *Let π_θ be an L_π-Lipschitz function with respect to θ, let M be the maximum completion length, and let $\tau>0$ be the Gumbel temperature. Assuming ϵ is independent of θ, the gradient of the reward term in Equation 8 is bounded as*

$$\|\nabla_\theta r(x,\hat{y})\| \;\le\; \frac{L_\pi}{\tau}\sum_{t=1}^{M}\left\|\frac{\partial r(x,\hat{y})}{\partial\hat{y}_t}\right\| \quad .$$

Proof. The gradient of the reward term can be written as:

$$\nabla_\theta r(x,\hat{y})=\sum_{t=1}^{M}\left(\frac{\partial\hat{y}_t}{\partial\theta}\right)^{\top}\frac{\partial r(x,\hat{y})}{\partial\hat{y}_t} \quad , \tag{12}$$

where $\hat{y}_t=\mathrm{Softmax}\big((z_\theta^t+\epsilon_t)/\tau\big)$. Softmax is 1-Lipschitz, so $\|\partial\hat{y}_t/\partial\theta\|\le\tau^{-1}\|\partial z_\theta^t/\partial\theta\|\le L_\pi/\tau$ by Assumption 3. Applying the triangle inequality and bounding each term,

$$\|\nabla_\theta r(x,\hat{y})\|\le\sum_{t=1}^{M}\left\|\frac{\partial\hat{y}_t}{\partial\theta}\right\|\left\|\frac{\partial r(x,\hat{y})}{\partial\hat{y}_t}\right\|\le\frac{L_\pi}{\tau}\sum_{t=1}^{M}\left\|\frac{\partial r(x,\hat{y})}{\partial\hat{y}_t}\right\| \quad . \tag{13}$$

□

C Additional Experimental Details

For all experiments, we use gradient checkpointing, bf16 precision for both the policy and reward models, an Adam optimizer, and vLLM for generation. Where applicable, we use an average KL penalty instead of the sum to control any potential length biases.

C.1 IMDb

We fully finetune all parameters in the `GPT2-small` model for this task, and anneal the Gumbel temperature τ from 1 to 0.2 following an exponential schedule. The training rewards reported in the main text average the final 200 training rewards of training.

Hyperparameter	GRPO/REINFORCE	TAG-BPO
Learning rate	$[10^{-4}, 10^{-5}, 10^{-6}]$	
Temperature	1	
Effective batch size	16	
Generations per prompt	2	
Training epochs	1	
Max completion length	16 tokens	
KL coefficient β	$[0, 0.01, 0.03, 0.05, 0.1]$	$[0, 1, 2, 3, 4]$

Table 4: Hyperparameters for the IMDb environment.

C.2 TL;DR

For additional details on how the reward model and SFT policies were trained, refer to Noukhovitch et al. (2025) and Huang et al. (2024). All policies were trained using LoRA (Hu et al., 2021) at rank $r = 16$ and $\alpha = 32$, applying adapters to all attention and feed-forward projection matrices in every transformer block. Evaluation uses a held-out dataset of prompts and human completions containing 6447 examples. Generation from the finetuned policies set the temperature to 1, and a single generation is evaluated for all 6447 examples. The generated completion and human completion are scored with the gold 6.9B parameter reward model, and the win-rate reports how often the generated completions from the policy outscore the human completion. The mean win-rate over the entire dataset is then reported for each seed, and the results report the mean and standard deviation of the five seeds. All reported KL divergences are computed against the same completions used for evaluation on the held-out dataset. The training rewards reported in the main text average the final 500 training rewards of training.

Hyperparameter	GRPO/REINFORCE	TAG-BPO
Learning rate	$[3 \times 10^{-6}, 10^{-5}, 3 \times 10^{-5}]$	
Temperature	1	
Effective batch size	[8, 16, 32]	
Generations per prompt	2	
Training epochs	1	
Max completion length	53 tokens	
KL coefficient β	$[0, 0.01, 0.03, 0.05, 0.1]$	$[0, 1, 2, 3, 4]$

Table 5: Hyperparameters for the TL;DR environment.

C.3 GSM8K

We finetune our policies with LoRA (Hu et al., 2021) at rank $r = 16$ and $\alpha = 32$, applying adapters to all attention and feed-forward projection matrices in every transformer block. Since the Skywork reward model uses a different <eos> token ID than the base policy in this task, it is not so straightforward to obtain rewards $r(x, y_{<l})$ for all $1 \leq l < L$ given $r(x, y_{<L})$. Therefore, we use the score function variant of TAG-BPO shown in Appendix B.3 for these experiments with a coefficient of 0.05 on the first term. Evaluations in this task are done by sampling 2 completions for each of the 1319 held out prompts, and reporting the average accuracy of the completions given by a correctness function. The training rewards reported in the main text average the final 100 training rewards of training.

We use the following prompt for all GSM8K experiments, including the toy experiment on the arbitrary target tokens:

GSM8K Prompt

```
<|im_start|>system

You are a helpful math assistant. Respond in the following format concisely within 50
 words, where
the answers are only a single number:

<reasoning>
...
</reasoning>
<answer>
...
</answer>
<|im_end|>
<|im_start|>user
What is the largest single-digit prime number?<|im_end|>
<|im_start|>assistant
<reasoning>
9 is divisible by 3 and 8 is divisible by 2, but 7 is prime.
</reasoning>
<answer>
7
</answer>
<|im_end|>
<|im_start|>user
<QUESTION>
<|im_end|>
<|im_start|>assistant
```

Hyperparameter	**Value**
Policy	Qwen3-0.6B-Base
Reward model	Skywork-Reward-V2-Qwen3-0.6B
Temperature	1
Learning rate	3×10^{-6}
Effective batch size	8
Generations per prompt	2
Training Steps	2000
Max completion length	256 tokens
KL coefficient β	0

Table 6: Hyperparameters for GRPO, REINFORCE, and TAG-BPO.

C.3.1 GSM8K Correctness Function

We use a correctness function that is mostly agnostic to the formatting unlike the standard one defined in Ackermann et al. (2026) and Hugging Face (2025). In general, it checks for numbers within the correct `<answer>` tags, or `\boxed{}` or if those fail, just the last number in the completion. Pseudocode for this checker is presented in Algorithm 1

C.4 Training Costs

All training costs are measured with the same batch size, number of generations, gradient accumulation steps and number of training steps across all methods. All reported values were measured on a single L40S GPU. The time measures the total wall-clock time of training, and memory displays the peak memory usage throughout training.

D Toy Termination Environment

In order to better understand the necessity of differentiating through the terminal condition, consider a simple language generation task where the goal is to maximize the cumulative sum of the token IDs in the generated prompt-response pair. We limit the vocabulary of the policy to the token

Algorithm 1 ANSWERCORRECT

Require: completion y, ground-truth answer $a^\star$
Ensure: $r \in \{0, 1\}$
$y \leftarrow$ prefix of y before <|im_end|>
$N(x) :=$ all matches in x of -?\d[\d,]*(?:\.\d+)?
if <answer> $\in y$ **and** $N($text inside the last <answer>$) \neq \varnothing$ **then**
    $\hat{a} \leftarrow$ first element of that set
else if y contains \boxed{b} **and** $N(b_{\text{last}}) \neq \varnothing$ **then**
    $\hat{a} \leftarrow$ last element of $N(b_{\text{last}})$
else if $N(y) \neq \varnothing$ **then**
    $\hat{a} \leftarrow$ last element of $N(y)$
else
    return 0
end if
return $\left|\text{TOFLOAT}(\text{STRIPCOMMAS}(\hat{a})) - a^\star\right| < 10^{-6}$

	IMDb		TL;DR		GSM8K	
Method	Time (min)	Mem (GB)	Time (min)	Mem (GB)	Time (min)	Mem (GB)
TAG-BPO	50	25	549	34	53	27
GRPO	42	22	416	31	44	20
REINFORCE	42	22	414	31	43	20

Table 7: Training costs for Table 2 for different methods. We show the average costs across 5 seeds for all methods.

IDs {0, 1, 2, 3}, set the maximum generation length to 3, and create a dataset of prompts $\mathcal{D}$ containing one of two possible prompts: either the single token ID 1 or 2 for a little added noise. Importantly, we set the <eos> condition to the token ID 3, and include it in reward calculations. The optimal response y is to always output the IDs (2,2,3), resulting in a total reward of either 8 or 9 depending on the sampled prompt; 2 is the highest value token that does not prematurely terminate the sequence and the final 3 is worth generating, since the maximum generation length has been reached anyway.

We train a policy on this task given the ground truth differentiable reward function using our terminal aware gradient introduced in Equation 7, TAG-BPO, and the Naive-BPO gradient from Equation 5. We also train a policy with GRPO as an additional unbiased baseline. The results shown in Figure 5 highlight the importance of marginalizing over the <eos> token. The terminal condition is not differentiable in the reward function, thus a naive Gumbel Softmax policy gradient lacks the understanding that earlier 3s actively truncate the response despite its immediate higher value.

E IMDB FULL KL ABLATION

Similar to the results shown in Figure 3, we also report an ablation of switching KL estimators for GRPO and BPO for the IMDb task in Figure 6.

F GSM8K PERPLEXITY

Following Yue et al. (2025), we measure the median [Q1, Q3] perplexity that the shared base policy π_0 assigned to completions made by the trained GRPO and TAG-BPO policies returned 1.45 [1.35, 1.60] and 2.01 [1.67, 2.51] respectively against 3.81 [2.41, 7.69] for the base completions. The perplexities were evaluated using 2 completions on the 1319 held out prompts.

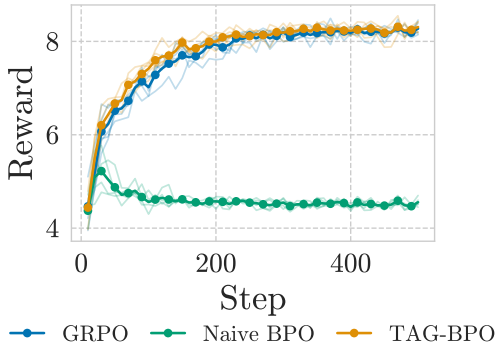

Figure 5: **A terminal aware policy gradient is necessary for the Gumbel Softmax gradient to be unbiased.** Policies are trained over five seeds with GRPO, Naive-BPO, and TAG-BPO for 500 gradient steps on the toy terminating cumulative sum environment.

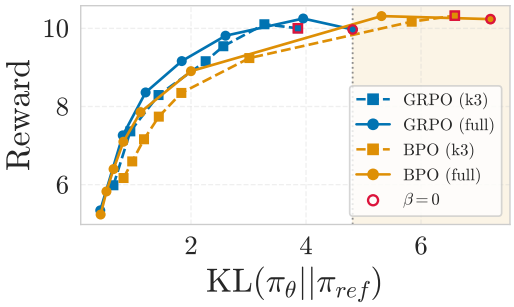

Figure 6: KL–reward trade-offs on IMDb comparing the `K3` and `KFull` estimators for both TAG-BPO and GRPO.