COMPUTATIONALLY EFFICIENT GRADIENT PROXIES FOR LARGE REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Gradient-based optimization supports applications throughout the development and deployment of large reasoning models (LRMs), but repeatedly obtaining gradients through explicit reasoning traces can be expensive. Inspired by recent advances in efficient reasoning, we introduce computationally efficient gradient proxies using shorter reasoning traces to obtain useful optimization guidance for the original model. We instantiate three proxies by omitting explicit reasoning, reasoning in continuous latent space, or combining these two approaches. We assess the quality and cost of these gradient proxies and evaluate their performance in four representative downstream tasks across stages of model development and deployment, including data preparation, training, task adaptation, and inference. Extensive experiments demonstrate improved performance with lower computational cost in various tasks, supporting the generality of gradient proxies across these distinct optimization problems.

1 INTRODUCTION

Gradient information plays a central role in optimizing large language models (LLMs), with applications extending beyond parameter updates. It guides the selection of supervision (Xia et al., 2024; Wang et al., 2026b), the construction of perturbations for adversarial training (Xhonneux et al., 2024; Sheshadri et al., 2025), task-prompt optimization (Wen et al., 2023; Das et al., 2025), and inference-time refinement (Dathathri et al., 2020; Wang et al., 2026a). This work focuses on large reasoning models (LRMs) that generate an explicit chain of thought (CoT) before the final answer, using intermediate reasoning steps as in CoT prompting (Wei et al., 2022). For methods that obtain gradients by generating and differentiating through reasoning traces, repeated optimization can incur substantial generation and backpropagation costs (Das et al., 2025; Wang et al., 2026a).

The cost of obtaining these gradients depends in part on how an optimization method handles the model's reasoning trace. For example, GReaTer (Das et al., 2025) first generates a trace and then differentiates the answer loss with respect to the prompt while holding the trace fixed. This yields a gradient conditional on the generated reasoning, without accounting for how a prompt change might alter that reasoning. Other formulations sample reasoning paths to estimate learning gradients (Phan et al., 2023; Yao et al., 2025), optimize prompts over sampled responses (Geisler et al., 2025), or relax discrete decoding to permit backpropagation (Goyal et al., 2017; Wichers et al., 2024). Query-based estimators infer input directions by evaluating generated responses under input perturbations (Hu et al., 2024). Although these formulations target different objectives, they share a major source of overhead: repeatedly generating and processing reasoning traces. Long traces also require more activation memory during backpropagation, making repeated gradient estimation particularly costly for LRMs.

Recent advances in efficient reasoning offer concrete ways to shorten this computation. Explicit thinking can be omitted: Qwen3 (Yang et al., 2025) supports thinking and non-thinking modes within the same model, and prior work shows that bypassing the explicit thinking phase can remain effective, particularly on simpler problems or under limited token budgets (Ma et al., 2025; Zhang et al., 2025). Other approaches compress intermediate reasoning: COCONUT (Hao et al., 2025) performs reasoning in continuous hidden states, while CODI (Shen et al., 2025) distills explicit CoT into a compact latent reasoning process. These approaches yield a lower computational cost compared with an explicit CoT model through shared parameters or reasoning distillation. However,

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.

it remains unclear whether *computationally cheaper gradients obtained through these methods can effectively guide optimization of the original CoT model*.

We introduce *computationally efficient gradient proxies* that guide optimization of the original CoT model through shorter reasoning computations. We develop three variants: direct answering (**GP-DA**) omits intermediate reasoning and computes gradients through a shorter answer-generation path; latent reasoning (**GP-LR**) uses CODI to distill explicit CoT into a separate model with shorter continuous latent states; a hybrid strategy (**GP-Hybrid**) combines proxy gradients computed with the target's current parameters via GP-DA and distilled reasoning via GP-LR. We use these proxy gradients to guide downstream optimization of the original model.

We conduct comprehensive evaluations on these gradient proxies, including the gradient quality, the computational cost and the performance in downstream tasks. We first compare them with a computationally expensive high-query reference, measuring directional agreement, target-model objective improvement, and computational cost. We then examine four applications as case studies across the LLM lifecycle: selecting supervision during data preparation, constructing adversarial perturbations for safety alignment, optimizing reusable task prompts, and guiding response search at inference time. The results show that gradient proxies can substantially reduce the computational cost while matching or improving downstream performance. The preferred proxy varies across applications: GP-DA offers the largest computational savings, while GP-LR and GP-Hybrid often provide stronger downstream performance.

Our contributions are threefold: **(1)** we propose several gradient proxies, including GP-DA, GP-LR, and GP-Hybrid, to bypass or compress explicit reasoning to guide the optimization on the original CoT model; **(2)** we demonstrate substantial computational savings of using gradient proxies while maintaining competitive performance in the downstream tasks, establishing an improved trade-off between computational cost and model utility; and **(3)** we evaluate these proxies in four representative downstream case studies, including selective SFT, safety alignment, prompt optimization, and test-time reasoning, to demonstrate their applicability across the LLM lifecycle.

2 Related Work

Gradient Applications in LLMs. Beyond parameter learning, gradients guide instruction-data selection (Xia et al., 2024), selective reasoning supervision (Wang et al., 2026b), and adversarial training (Zhu et al., 2020; Sheshadri et al., 2025). They also support prompt optimization (Wen et al., 2023; Das et al., 2025), controlled generation (Dathathri et al., 2020), and test-time reasoning (Wang et al., 2026a). These applications optimize different variables but require informative gradients.

Gradient Estimation in LLMs. Discrete generation complicates differentiation through reasoning. GReaTer conditions answer-loss gradients on a generated trace (Das et al., 2025), while other methods use relaxations or surrogate backward rules (Wichers et al., 2024; Wang et al., 2026a). Sampling-based methods estimate learning gradients over reasoning trajectories (Phan et al., 2023; Yao et al., 2025), and query-based methods estimate input sensitivities from perturbed evaluations (Hu et al., 2024). Our proxies instead shorten the reasoning computation used to obtain gradient guidance.

Surrogate Models for Gradient Estimation. Surrogate gradients guide query-based adversarial attacks (Cheng et al., 2019) and offline black-box optimization (Hoang et al., 2024). ZeroGrads learns local surrogate losses for graphics pipelines (Fischer & Ritschel, 2024), while GradPIE encourages gradient alignment through path-integral consistency (Momeni et al., 2025). For LLM data selection, IProX builds smaller proxies through low-rank compression and gradient–logit alignment (Chen et al., 2026). These methods emphasize useful optimization signals beyond output prediction. We omit or compress reasoning for cheaper gradients and evaluate their utility on the original model.

3 Computationally Efficient Gradient Proxies

3.1 Preliminaries

Given an input x, a CoT model $M_{\boldsymbol{\theta}}$ generates a sequence of intermediate reasoning tokens $\boldsymbol{z} = (z_1, \ldots, z_L)$ and then an answer in an autoregressive manner, where each token belongs to the

vocabulary $\mathcal{V}$. Since the answer is generated in the context of the reasoning trace, changes in the input x or parameters $\boldsymbol{\theta}$ can affect the final answer through changes in the reasoning trace.

There are many algorithms using gradients from a large reasoning model to optimize a continuous variable $\boldsymbol{u} \in \mathbb{R}^m$, such as input perturbations in adversarial generation (Xhonneux et al., 2024), prompt embeddings in gradient-based prompt search (Wen et al., 2023), hidden states in controllable text generation (Dathathri et al., 2020), and continuous token logits in constrained decoding (Qin et al., 2022). The gradient-based algorithms support a broad range of applications, including robustness enhancement, model adaptation, controllable generation, and constrained decoding. We write $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ for the task loss, suppressing the fixed input x. The reasoning trace may depend on $\boldsymbol{u}$, denoted $\boldsymbol{z} = \boldsymbol{z}(\boldsymbol{u})$. The algorithm first estimates the gradient $\nabla_{\boldsymbol{u}}\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ and then optimizes $\boldsymbol{u}$ by gradient-based methods. With a slight abuse of notation, we use $\nabla_{\boldsymbol{u}}\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ to also denote a practical gradient estimate when the underlying computation is not differentiable.

It is usually difficult to obtain a numerically reliable and computationally affordable gradient $\boldsymbol{g} := \nabla_{\boldsymbol{u}}\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ for large reasoning models, posing challenges for a broad range of downstream tasks. First, reasoning traces $\boldsymbol{z}$ are typically long, incurring considerable overhead in both computational complexity and memory consumption. Second, the discrete nature of autoregressive decoding renders the overall process theoretically non-differentiable. For instance, greedy decoding selects the next token z_{t+1} in the reasoning trace by $z_{t+1} = \arg\max_{1 \le j \le |\mathcal{V}|} \left(M_{\boldsymbol{\theta}}(\boldsymbol{u}; x, \boldsymbol{z}_{1:t})\right)_j$. Hard selection is locally constant when the maximum logit is unique and discontinuous at maximum-switching boundaries. In both cases, ordinary backpropagation cannot capture the effect of token changes on subsequent reasoning. Beyond greedy decoding, other decoding schemes exhibit similar undesirable properties in the token-selection operation.

Existing methods handle discrete reasoning in three major ways. Fixed-trace differentiation generates fixed reasoning traces and then treats them as constant input in gradient computation (Das et al., 2025). Differentiable relaxations and straight-through estimators (STE) instead replace hard selection with continuous approximations or substitute a backward rule for the discrete operation (Bengio et al., 2013; Goyal et al., 2017; Wang et al., 2026a). Sampling-based approaches estimate gradients by averaging over reasoning trajectories, as studied in latent-variable CoT training and gradient-variance-aware sampling (Phan et al., 2023; Yao et al., 2025). The formulations of these three approaches are given in Appendix A.1. However, when reasoning traces are long, all these methods become computationally expensive. Moreover, longer reasoning traces mean more decoding steps and thus more approximations. The resulting approximation errors in these methods can propagate, yielding less accurate gradients. This work aims to address these concerns.

3.2 Computationally Efficient Gradient Estimation

We address the computational cost of gradient calculation in large reasoning models through a shorter proxy computation, and use the proxy gradient to guide optimization of the variable $\boldsymbol{u}$ in the original model $M_{\boldsymbol{\theta}}$. The proxy model $M_{\widehat{\boldsymbol{\theta}}}$ is designed to supply useful update directions efficiently without reproducing the target model's full reasoning trace. We obtain these shorter computations by fine-tuning: GP-DA adds a non-thinking mode to the target itself, while GP-LR distills a separate latent-reasoning model. We use $\widehat{\boldsymbol{z}} = (\widehat{z}_1, \ldots, \widehat{z}_{L'})$, with $L' \ll L$, to denote the shorter trace. The fine-tuning aims to enforce $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) \simeq \mathcal{L}_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{u}, \widehat{\boldsymbol{z}})$ to make $M_{\widehat{\boldsymbol{\theta}}}$ reproduce the behavior of $M_{\boldsymbol{\theta}}$ using a much shorter reasoning trace. We use the resulting gradient $\nabla_{\boldsymbol{u}}\mathcal{L}_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{u}, \widehat{\boldsymbol{z}})$ to guide optimization of $\boldsymbol{u}$, and evaluate its agreement with a reference gradient and its utility on the original model. We introduce three proxy constructions below. Their common gradient coordinates and combination are detailed in Appendix A.2.

Gradient Proxy with Direct-Answering (GP-DA). Intensive evidence indicates that a reasoning model can retain useful task capabilities when its explicit thinking phase is omitted (Ma et al., 2025; Yang et al., 2025). GP-DA fine-tunes the CoT model to support direct answering with no intermediate trace, i.e., $\widehat{\boldsymbol{z}} = \emptyset$. The target and GP-DA share the resulting parameters, $\widehat{\boldsymbol{\theta}}_{\text{DA}} = \boldsymbol{\theta}$, and differ in reasoning mode. GP-DA therefore tracks subsequent target updates while avoiding autoregressive reasoning generation. We use $\mathcal{L}_{\widehat{\boldsymbol{\theta}}_{\text{DA}}}(\boldsymbol{u}, \widehat{\boldsymbol{z}})$ and $\widehat{\boldsymbol{g}}_{\text{DA}} := \nabla_{\boldsymbol{u}}\mathcal{L}_{\widehat{\boldsymbol{\theta}}_{\text{DA}}}(\boldsymbol{u}, \widehat{\boldsymbol{z}})$ to represent the loss objective function and the gradient in GP-DA, respectively.

Gradient Proxy with Latent-Reasoning (GP-LR). GP-LR retains a reasoning trace while shortening its path and removing the token-selection operation in the decoder. We instantiate it with CODI (Shen

et al., 2025), which distills explicit reasoning into continuous states using answer supervision and hidden-state alignment. Specifically, CODI is based on the COCONUT architecture (Hao et al., 2025), which enables the model to reason in a continuous latent space. Here, $\widehat{\boldsymbol{z}}$ denotes a set of continuous latent representations rather than tokens, and the decoder generates the reasoning trace without a token-selection operation via $\widehat{z}_{t+1} = M_{\widehat{\boldsymbol{\theta}}}(\boldsymbol{u}; x, \widehat{\boldsymbol{z}}_{1:t})$. Compared with traditional decoding schemes, the reasoning traces generated by COCONUT are much shorter, and the entire reasoning generation process is fully differentiable, since COCONUT iterates in a continuous latent space using compressed dense representations rather than discrete tokens. In summary, by applying CODI to distill reasoning traces in a COT model, GP-LR manages to reproduce the reasoning capabilities in a much shorter and fully differentiable path. Therefore, it improves the computational efficiency and addresses the non-differentiability issue for gradient calculation. We use $\mathcal{L}_{\widehat{\boldsymbol{\theta}}_{\text{LR}}}(\boldsymbol{u}, \widehat{\boldsymbol{z}})$ and $\widehat{\boldsymbol{g}}_{\text{LR}} := \nabla_{\boldsymbol{u}} \mathcal{L}_{\widehat{\boldsymbol{\theta}}_{\text{LR}}}(\boldsymbol{u}, \widehat{\boldsymbol{z}})$ to represent the loss objective function and the gradient in GP-LR, respectively.

GP-Hybrid. GP-DA uses the exact same parameters as the COT model but without reasoning, while GP-LR has reasoning compatibility but different model parameters. GP-Hybrid interpolates their gradients. With mixing coefficient $\lambda \in [0, 1]$, its estimate is $\widehat{\boldsymbol{g}}_{\text{HY}} = \lambda \widehat{\boldsymbol{g}}_{\text{DA}} + (1 - \lambda)\widehat{\boldsymbol{g}}_{\text{LR}}$. The computational complexity of GP-DA is much lower than that of GP-LR, so GP-Hybrid has a similar computational complexity to GP-LR.

Figure 1 summarizes the workflow of GP-DA, GP-LR and GP-Hybrid.

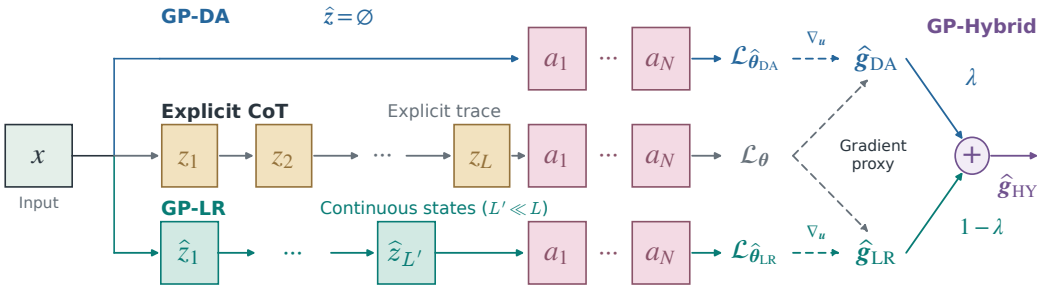


Figure 1: Gradient proxies for explicit CoT. The target uses the reasoning trace $\boldsymbol{z} = (z_1, \ldots, z_L)$; GP-DA uses $\widehat{\boldsymbol{z}} = \emptyset$, and GP-LR uses a short continuous trace $\widehat{\boldsymbol{z}} = (\widehat{z}_1, \ldots, \widehat{z}_{L'})$, with $L' \ll L$. The losses are denoted by $\mathcal{L}_{\boldsymbol{\theta}}$, $\mathcal{L}_{\widehat{\boldsymbol{\theta}}_{DA}}$, and $\mathcal{L}_{\widehat{\boldsymbol{\theta}}_{LR}}$, with their arguments omitted for clarity; $(a_1, \ldots, a_N)$ denote answer tokens. The three proxy computations illustrate GP-DA, GP-LR, and GP-Hybrid.

3.3 The Quality and Computational Cost of the Estimated Gradients

To assess whether our proxies provide accurate and useful gradient estimates at lower computational cost, we compare their agreement with a high-query target model reference, their effect on the target objective, and their running time.

High-Query Reference. Let $F(\boldsymbol{u}) = \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}(\boldsymbol{u}))$, where the target generates a trace $\boldsymbol{z}(\boldsymbol{u})$ at each input. Following Nesterov & Spokoiny (2017), we estimate the gradient of its Gaussian-smoothed objective using two-sided finite differences. For $\boldsymbol{u} \in \mathbb{R}^m$, we sample K independent directions $\boldsymbol{v}_i \sim \mathcal{N}(0, \mathbf{I}_m)$ and use $2K$ evaluations:

$$\widehat{\boldsymbol{g}}_{\text{ref}} = \frac{1}{K} \sum_{i=1}^{K} \frac{F(\boldsymbol{u} + \sigma \boldsymbol{v}_i) - F(\boldsymbol{u} - \sigma \boldsymbol{v}_i)}{2\sigma} \boldsymbol{v}_i. \tag{1}$$

Here, σ sets the smoothing scale, and each evaluation regenerates the trace. Appendix B details the reference construction and choice of K.

Task Objective. To test whether an estimated gradient leads to useful updates in the original model, we consider targeted attacks on its input embeddings as a simple example task. Here, $\boldsymbol{u}$ contains the question-token embeddings; the question text and model parameters $\boldsymbol{\theta}$ are fixed. The loss objective $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ is the mean token-wise negative log-likelihood of a specified target answer. In each iteration, we calculate the gradient by high-query references, baselines or proxy gradient estimation.

Table 1: Gradient quality and estimation cost on R1-Distill-Qwen-1.5B.

	Jailbreak			GSM8K		
Estimator	Cos.↑	ΔJ ↑	Time↓	Cos.↑	ΔJ ↑	Time↓
High-query reference	n/a	0.359	558.307	n/a	0.45906	284.042
Teacher-Forcing	0.489	0.216	0.188	**0.2664**	0.17713	0.089
STE	0.028	0.053	25.606	0.1390	0.15125	16.178
GP-DA	0.306	0.168	**0.051**	0.0828	0.16073	**0.087**
GP-LR	0.470	0.244	1.524	0.2472	**0.26129**	3.350
GP-Hybrid	**0.510**	**0.254**	1.579	0.2545	0.24071	3.423

Cos.: cosine similarity to the high-query reference; ΔJ: target-loss reduction; time: seconds per example. All entries are means. Bold marks the best non-reference result; n/a: not applicable.

We compute the perturbations $\delta = -R \cdot \boldsymbol{g}/\|\boldsymbol{g}\|_2$ based on the obtained gradient $\boldsymbol{g}$ with a common radius R, and then evaluate the loss using the perturbed input embeddings. A lower loss indicates a higher likelihood of the target answer, which is the task objective.

Experimental setup. We evaluate R1-Distill-Qwen-1.5B (Guo et al., 2025) on Jailbreak and GSM8K (Cobbe et al., 2021), using $K = 1024$ perturbation pairs for the reference $\widehat{\boldsymbol{g}}_{\text{ref}}$. We compare GP-DA, GP-LR, and GP-Hybrid with Teacher-Forcing, adapted from GReaTer's fixed-trace differentiation (Das et al., 2025), and STE (Bengio et al., 2013). Implementations are described in Appendix C. Table 1 reports mean cosine similarity with $\widehat{\boldsymbol{g}}_{\text{ref}}$, mean target-loss reduction, and mean runtime per example. We regenerate the target model's reasoning at each input and calculate $\Delta J = F(\boldsymbol{u}) - F(\boldsymbol{u} + \delta)$, so a positive value indicates improvement. Runtime in Table 1 covers gradient estimation only, excluding proxy model training and evaluation.

Results. GP-DA has the lowest runtime on both datasets, with a cost close to Teacher-Forcing on GSM8K. Retaining latent reasoning yields larger reduction in target loss: GP-Hybrid performs best among the non-reference estimators on Jailbreak, while GP-LR leads on GSM8K. Agreement with the reference and improvement of the target objective capture different aspects of gradient quality. We evaluate how these proxy gradients support downstream applications in the next Section.

4 Case Studies: Gradient Proxies across the LLM Lifecycle

4.1 Overview and Experimental Setup

The four applications in Figure 2 use gradients to make different decisions at different stages of the LLM lifecycle: which reasoning segments to pick in data preparation (Section 4.2), which input perturbations to train against in model alignment (Section 4.3), which prompt tokens to edit in task adaptation (Section 4.4), and which reasoning candidates to generate during inference (Section 4.5). In these tasks, the continuous variable $\boldsymbol{u}$ being optimized may have different concrete meanings, but the popular methods all require gradients $\nabla_{\boldsymbol{u}}\mathcal{L}(\boldsymbol{u}, \boldsymbol{z})$ for optimization. We use task-specific checkpoints of R1-Distill-Qwen-1.5B (R1) (Guo et al., 2025) and Llama-3.1-8B-Instruct (Llama) (Grattafiori et al., 2024). Unless otherwise specified, GP-LR uses $L' = 6$ and GP-Hybrid uses $\lambda = 0.5$.

4.2 Data Preparation: Selective Supervised Fine-Tuning

Task Objective. Given a question x, a segmented reasoning trace $\boldsymbol{z}$, and a target answer a^*, segment-selective SFT (Wang et al., 2026b) scores each segment by its contribution to predicting a^* and selects segments for supervision. Scoring occurs during data preparation with fixed model parameters.

To obtain these scores, we represent the trace by its token embeddings $\boldsymbol{u}$ and define the target-answer loss as $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) = -\log p_{\boldsymbol{\theta}}(a^* \mid \boldsymbol{u}, \boldsymbol{z})$. Following Wang et al. (2026b), we compute segment scores from this gradient or a proxy gradient and select supervision for SFT. GP-Hybrid interpolates DA/LR gradients before attribution.

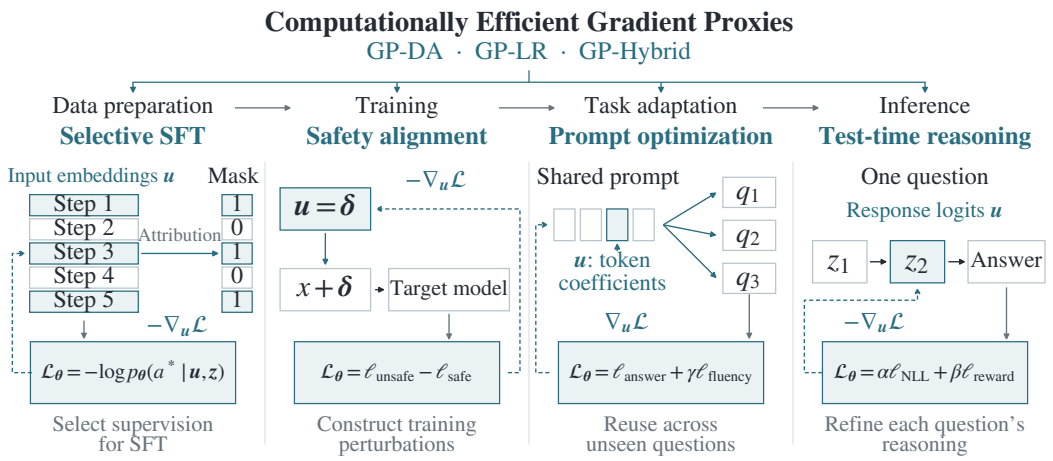


Figure 2: Four uses of gradient proxies: selecting supervision, constructing training perturbations, optimizing a reusable task prompt, and refining an individual question's reasoning.

We compare original and proxy gradients for supervision selection and fine-tuning on LIMO (Ye et al., 2025). We evaluate the fine-tuned models on MATH-500 (Lightman et al., 2024), AMC23, and AIME24 (Li et al., 2024) for in-domain accuracy, and GPQA-Diamond (Rein et al., 2024), Minerva (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024) for out-of-domain accuracy. We report gradient-estimation time cost relative to Selective SFT. See Appendix D.1 for details.

Results. Replacing full-trace gradients with proxy gradients makes supervision selection substantially cheaper and can improve the resulting model's accuracy (Table 2). GP-DA provides the largest saving, using only about 13% of the original computation time while improving in-domain accuracy on both models. Including latent reasoning through GP-LR or GP-Hybrid yields the strongest downstream performance: Hybrid achieves the best in-domain average on R1, while LR leads out of domain on R1 and in both domains on Llama. These gains still require 41.1–74.8% less computation time than the original Selective SFT method, with greater relative savings on the larger Llama model.

4.3 Training: Safety Alignment

Task Objective. Adversarial training (AT) for LLM safety (Xhonneux et al., 2024) trains models to respond safely to adversarial inputs. For each example, an inner optimization constructs a perturbed input, and an outer update trains the model on safe responses to both clean and perturbed inputs.

To construct these adversarial inputs, we optimize an additive perturbation $\boldsymbol{u} = \delta$ to the question embeddings under the objective $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) = \ell_{\text{unsafe}} - \ell_{\text{safe}}$, where ℓ_{unsafe} and ℓ_{safe} are the mean token negative log-likelihoods of the unsafe target answer and the safe reference answer, respectively. Following the original AT procedure, we update $\boldsymbol{u}$ along $-\nabla_{\boldsymbol{u}}\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ or its proxy estimate, subject to a perturbation budget, to make the unsafe answer more likely relative to the safe response.

We train on STAR-1 (Wang et al., 2026c), retaining the same outer training procedure across methods. We evaluate models using StrongREJECT scores (Souly et al., 2024) and AutoRAN ASR@7 (Liang et al., 2026) for attack resistance, XSTest over-refusal rates (Röttger et al., 2024) for refusals of benign requests, and accuracy on MATH-500 (Lightman et al., 2024) and GPQA-Diamond (Rein et al., 2024) for retained reasoning capability. We also report gradient-estimation time cost relative to Teacher Forcing AT. See Appendix D.2 for details.

Results. GP-Hybrid improves the balance between attack resistance and reasoning capability (Table 3), achieving the lowest StrongREJECT score and AutoRAN ASR on both models while retaining MATH accuracy close to Standard SFT. On R1, it retains 61.8% MATH accuracy, compared with 56.2% for Teacher Forcing AT, despite stronger attack resistance. These improvements also come at a lower gradient-estimation cost, while GP-DA provides the largest computational saving. All three proxies achieve greater relative time savings on R1, consistent with its longer reasoning traces. We analyze the mixing weight in Appendix D.2.1.

Table 2: Selective SFT: accuracy (%, ↑) and Time Cost (%, ↓), normalized to the original Selective SFT method (100%) for each model. Avg. Token Length refers to the reasoning traces used for timing. Avg. is the unweighted mean within each domain; GPQA denotes GPQA-Diamond and Olympiad denotes OlympiadBench.

Method	In-domain				Out-of-domain				Time Cost ↓
	MATH500	AMC23	AIME24	Avg.	GPQA	Minerva	Olympiad	Avg.	
R1-Distill-Qwen-1.5B (Avg. Token Length: 5674.5)									
Full-CoT SFT	75.0	57.5	20.0	50.8	24.7	18.8	35.4	26.3	–
Selective SFT	76.8	50.0	23.3	50.0	23.7	18.0	39.7	27.1	100.0
- w/ GP-DA	76.8	**62.5**	20.0	53.1	26.8	**23.5**	38.4	29.6	**13.2**
- w/ GP-LR	76.8	60.0	20.0	52.3	**28.3**	22.8	**41.6**	**30.9**	45.7
- w/ GP-Hybrid	**77.4**	**62.5**	**26.7**	**55.5**	27.3	**23.5**	38.1	29.6	58.9
Llama-3.1-8B-Instruct (Avg. Token Length: 5234.3)									
Full-CoT SFT	48.0	22.5	0.0	23.5	30.8	15.1	19.0	21.6	–
Selective SFT	44.6	27.5	0.0	24.0	29.8	16.5	19.4	21.9	100.0
- w/ GP-DA	45.6	**30.0**	0.0	25.2	28.8	16.2	**19.7**	21.6	**13.0**
- w/ GP-LR	**48.8**	27.5	**3.3**	**26.5**	**35.4**	**18.4**	18.4	**24.1**	25.2
- w/ GP-Hybrid	47.4	27.5	0.0	25.0	32.3	17.3	17.5	22.4	38.1

Table 3: Safety AT: attack resistance, reasoning capability, over-refusal, and Time Cost. SR is the mean StrongREJECT score (0–1); XS ORR is the XSTest over-refusal rate; AutoRAN reports ASR@7. All other values are percentages. Time Cost is normalized to Teacher Forcing AT (100%) for each model. Avg. Token Length refers to the reasoning traces used for timing.

Method	Safety ↓		Capability ↑		XS ORR ↓	Time Cost ↓
	SR score	AutoRAN	MATH	GPQA		
R1-Distill-Qwen-1.5B (Avg. Token Length: 1212.5)						
Standard SFT	0.05750	68.3	**62.4**	32.7	37.0	–
Teacher Forcing AT	0.05614	61.0	56.2	26.5	33.4	100.0
- w/ GP-DA	0.05705	61.7	60.2	24.5	33.0	**16.4**
- w/ GP-LR	0.05410	62.3	59.6	32.7	33.8	44.3
- w/ GP-Hybrid	**0.05156**	**59.1**	61.8	**35.7**	**29.8**	59.8
Llama-3.1-8B-Instruct (Avg. Token Length: 573.0)						
Standard SFT	0.03433	44.2	**43.8**	26.8	**12.0**	–
Teacher Forcing AT	0.03611	59.7	41.2	25.8	15.0	100.0
- w/ GP-DA	0.03403	43.5	40.4	27.3	14.2	**21.5**
- w/ GP-LR	0.03148	41.6	**43.8**	29.3	16.4	51.6
- w/ GP-Hybrid	**0.03125**	**41.3**	43.6	**30.8**	14.6	68.4

4.4 Task Adaptation: Prompt Optimization

Task Objective. GReaTer (Das et al., 2025) optimizes a shared prompt for a downstream task. Starting from a CoT prompt, it proposes token replacements and evaluates the edited prompts on an optimization set. The selected prompt is reused on unseen questions, with model parameters fixed.

We represent editable prompt positions by token-selection coefficients $\boldsymbol{u}$ and define the objective as $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) = \ell_{\text{answer}} + \gamma \ell_{\text{fluency}}$. Here, ℓ_{answer} is the mean token negative log-likelihood of the correct answer, while ℓ_{fluency} penalizes unlikely prompt tokens, with weight γ. Following GReaTer, we use $\nabla_{\boldsymbol{u}} \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ to rank candidate token replacements. Our proxies compute answer-loss gradients, retaining the target model's fluency objective and candidate evaluation.

We optimize prompts on GSM8K (Cobbe et al., 2021) and five BBH tasks (Suzgun et al., 2023): object counting, tracking shuffled objects, hyperbaton, causal judgement, and movie recommendation. We evaluate the selected prompts on held-out questions, reporting answer accuracy and gradient-estimation time cost relative to GReaTer. See Appendix D.3 for details.

Results. GP-LR achieves the highest average accuracy across the six tasks on both models while using only 26.9% and 40.0% of GReaTer's computation time on R1 and Llama, respectively (Table 4). Its average accuracy gains of 2.9 and 2.5 percentage points show that reducing gradient cost need not compromise prompt quality. GP-DA reduces cost further but lowers accuracy relative to GReaTer, suggesting that retaining latent reasoning helps identify more effective prompt edits. Llama produces shorter BBH outputs than R1, consistent with its smaller relative time savings.

Table 4: Prompt optimization: accuracy (%, ↑) and Time Cost (%, ↓), normalized to GReaTer (100%) on the same model. Avg. Token Length measures normal-inference outputs, including reasoning and the final answer. Avg. is the unweighted mean over GSM8K and the five BBH tasks: object counting (Count), tracking shuffled objects (Track), hyperbaton (Hyper.), causal judgement (Causal), and movie recommendation (Movie).

Method	GSM8K	BBH					Avg.	Time Cost ↓
		Count	Track	Hyper.	Causal	Movie		
R1-Distill-Qwen-1.5B (Avg. Token Length: 2991.1)								
Unoptimized CoT prompt	64.4	61.0	58.0	55.0	53.0	32.0	53.9	–
GReaTer	71.4	64.0	**77.0**	55.0	62.0	32.0	60.2	100.0
- w/ GP-DA	68.8	60.0	67.0	55.0	61.0	31.0	57.1	**13.8**
- w/ GP-LR	**71.6**	**68.0**	74.0	**60.0**	63.0	**42.0**	**63.1**	26.9
- w/ GP-Hybrid	71.0	62.0	66.0	56.0	**65.0**	37.0	59.5	42.7
Llama-3.1-8B-Instruct (Avg. Token Length: 291.9)								
Unoptimized CoT prompt	83.2	84.0	76.0	68.0	58.0	52.0	70.2	–
GReaTer	84.2	90.0	78.0	**69.0**	64.0	52.0	72.9	100.0
- w/ GP-DA	82.8	90.0	78.0	63.0	61.0	48.0	70.5	**21.6**
- w/ GP-LR	**84.3**	**93.0**	78.0	68.0	**65.0**	**64.0**	**75.4**	40.0
- w/ GP-Hybrid	84.1	91.0	**79.0**	68.0	**65.0**	51.0	73.0	82.6

4.5 Inference: Test-Time Reasoning

Task Objective. Test-time reasoning searches over candidate reasoning traces to improve an answer without updating model parameters. ∇-Reasoner (Wang et al., 2026a) guides this search with gradients, using a reward model to assess the generated responses.

We optimize response-token logits $\boldsymbol{u}$ using $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) = \alpha \ell_{\mathrm{NLL}} + \beta \ell_{\mathrm{reward}}$, following ∇-Reasoner. The likelihood term favors probable responses under the target model; the reward term favors higher scores. Proxy gradients guide candidate generation; GP-Hybrid combines DA and LR guidance.

We evaluate on GSM8K (Cobbe et al., 2021), AMC (Li et al., 2024), and MATH-500 (Lightman et al., 2024), reporting answer accuracy and end-to-end time cost relative to ∇-Reasoner. See Appendix D.4 for details.

Table 5: Test-time reasoning: accuracy (%, ↑) and end-to-end Time Cost (%, ↓). Time Cost is the unweighted mean of the three dataset-wise time ratios, normalized to ∇-Reasoner (100%) on the same model. Bold marks the highest accuracy and lowest Time Cost within each model.

Method	**R1-Distill-Qwen-1.5B**				**Llama-3.1-8B-Instruct**			
	GSM8K	AMC	MATH-500	Time Cost ↓	GSM8K	AMC	MATH-500	Time Cost ↓
Greedy	72.0	16.9	36.0	–	76.3	22.9	40.6	–
∇-Reasoner	81.3	30.1	46.6	100.0	84.1	24.1	55.8	100.0
- w/ GP-DA	83.4	28.9	46.6	**12.9**	86.6	28.9	58.6	**11.4**
- w/ GP-LR	83.4	**32.5**	**49.2**	29.1	86.4	**31.3**	58.4	20.3
- w/ GP-Hybrid	**83.5**	30.1	47.8	40.3	**86.9**	30.1	**58.8**	30.5

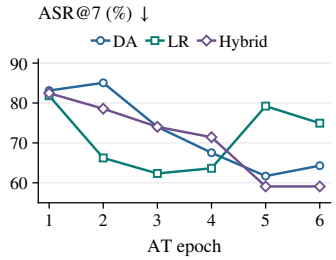
Results. GP-LR and GP-Hybrid match or exceed ∇-Reasoner's accuracy across all six model–dataset combinations with dataset-averaged end-to-end Time Cost of 20.3–40.3% (Table 5). GP-DA reduces this cost further to 11.4–12.9%, with lower accuracy only on R1's AMC evaluation. Among the

latent-reasoning variants, GP-LR costs less than Hybrid on both models and achieves the best AMC accuracy, whereas Hybrid performs best on GSM8K. Thus, Hybrid's additional computation is useful for some tasks but does not consistently improve upon LR guidance.

4.6 Ablation Studies

We examine proxy behavior during training, transfer across CODI training sources, and the effect of latent depth.

(a) Training dynamics



(b) CODI source

Accuracy (%) ↑

Source	SFT	TTR
LIMO	**77.4**	**84.3**
GSM8K	77.2	83.5
FOLIO	76.8	83.1
Mix	77.0	83.4

(c) Latent depth

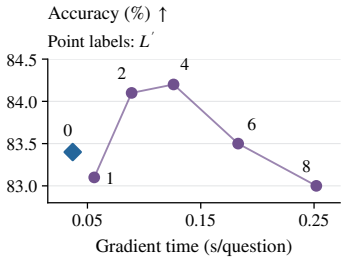


Figure 3: Ablations on R1. (a) AutoRAN ASR@7 during safety AT. (b) GP-Hybrid accuracy with different CODI training sources: selective SFT (SFT) on MATH-500 and test-time reasoning (TTR) on GSM8K. (c) GP-Hybrid accuracy versus proxy-gradient time on GSM8K.

Hybrid during training. We compare the three proxies across six safety AT epochs on R1 (Figure 3a). LR performs best in the first four epochs, while Hybrid leads in the final two, ending at 59.1% AutoRAN ASR versus 64.3% for DA and 74.9% for LR. Unlike the frozen LR proxy, DA shares the target's evolving parameters. Hybrid's stronger late-stage performance is consistent with DA complementing LR as the target diverges from LR's training model.

CODI transferability. We compare four CODI training sources using GP-Hybrid for selective SFT on MATH-500 and test-time reasoning on GSM8K (Figure 3b). Accuracy varies by only 0.6 and 1.2 percentage points, respectively. For test-time reasoning, LIMO-trained CODI even outperforms the task-matched GSM8K-trained proxy. Matching the proxy's training source to the downstream dataset therefore does not guarantee better guidance. The small variation suggests that a trained CODI proxy can be reused across these applications without substantial loss in downstream accuracy.

Latent depth. We vary CODI depth L' during training and inference for GP-Hybrid on GSM8K (Figure 3c). Four steps achieve the highest accuracy (84.2%), while two reach 84.1% at lower cost. Two steps reduce gradient time by approximately 29% for a difference of 0.1 percentage points in accuracy. Increasing depth beyond four raises cost and lowers accuracy, indicating that a longer latent computation does not necessarily provide more useful guidance.

5 Conclusion

We presented gradient proxies that obtain optimization guidance for large reasoning models through shorter reasoning computations. GP-DA removes explicit reasoning, GP-LR retains a compact latent trace, and GP-Hybrid combines their guidance. The three proxies offer different benefits: DA prioritizes efficiency, LR retains useful reasoning information, and Hybrid provides complementary guidance as the target changes during training. Our case studies cover four representative tasks in model development and deployment: selective SFT, safety alignment, prompt optimization, and test-time reasoning. The results show that these proxies reduce the cost of obtaining gradients while supporting effective optimization of the original model. Our ablations further examine their behavior across training stages, CODI training sources, and latent depths. Together, these findings support compact, reusable gradient proxies as a practical way to make gradient-based optimization of reasoning models more efficient.

AI USE STATEMENT

Generative AI assistance in this work included manuscript editing, literature searches, translation and analysis of existing literature, and code revision. All material prepared with AI assistance was reviewed by the authors. We have not used generative AI tools for proposing or refining hypotheses, assisting in the writing of proofs, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, designing or providing feedback on research methodology or experiments, supporting qualitative and thematic data analysis,or interpreting results. Generating synthetic data sets, cleaning and reformatting datasets, and transcribing recordings of research material are not applicable to this work. We take full responsibility for the accuracy and integrity of the manuscript, code, and accompanying artifacts, including all AI-assisted content.

REFERENCES

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. URL https://arxiv.org/abs/1308.3432.

Sirui Chen, Yunzhe Qi, Mengting Ai, Yifan Sun, Ruizhong Qiu, Jiaru Zou, and Jingrui He. Influence-preserving proxies for gradient-based data selection in LLM fine-tuning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/1916a6b025c7c618636f2c76dd8184cf-Abstract-Conference.html.

Shuyu Cheng, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Improving black-box adversarial attacks with a transfer-based prior. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/32508f53f24c46f685870a075eaaa29c-Abstract.html.

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. *arXiv preprint arXiv:2110.14168*, 2021.

Sarkar Snigdha Sarathi Das, Ryo Kamoi, Bo Pang, Yusen Zhang, Caiming Xiong, and Rui Zhang. GReaTer: Gradients over reasoning makes smaller language models strong prompt optimizers. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/18a42aad2fa8aa871e2ee20d425c208d-Abstract-Conference.html.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020.

Michael Fischer and Tobias Ritschel. ZeroGrads: Learning local surrogates for non-differentiable graphics. *ACM Transactions on Graphics*, 43(4):1–15, 2024. doi: 10.1145/3658173.

Simon Geisler, Tom Wollschläger, M. H. I. Abdalla, Vincent Cohen-Addad, Johannes Gasteiger, and Stephan Günnemann. REINFORCE adversarial attacks on large language models: An adaptive, distributional, and semantic objective. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 19082–19111, 2025. URL https://proceedings.mlr.press/v267/geisler25a.html.

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. Differentiable scheduled sampling for credit assignment. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 366–371, 2017. doi: 10.18653/v1/P17-2058.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

Minh Hoang, Azza Fadhel, Aryan Deshwal, Jana Doppa, and Trong Nghia Hoang. Learning surrogates for offline black-box optimization via gradient matching. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 18374–18393, 2024. URL https://proceedings.mlr.press/v235/hoang24a.html.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. Gradient cuff: Detecting jailbreak attacks on large language models by exploring refusal loss landscapes. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4011. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e46984e056185b21ddb1e7973c365f14-Abstract-Conference.html.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/18abbeef8cfe9203fdf9053c9c4fe191-Abstract-Conference.html.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. NuminaMath. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/AI-MO/NuminaMath-CoT.

Jiacheng Liang, Tanqiu Jiang, Yuhui Wang, Rongyi Zhu, Fenglong Ma, and Ting Wang. AutoRAN: Automated hijacking of safety reasoning in large reasoning models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 42897–42931, 2026. doi: 10.18653/v1/2026.acl-long.1988.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*, 2025. URL https://arxiv.org/abs/2504.09858.

Ali Momeni, Stefan Uhlich, Arun Venkitaraman, Chia-Yu Hsieh, Andrea Bonetti, Ryoga Matsuo, Eisaku Ohbuchi, and Lorenzo Servadei. Locality-aware surrogates for gradient-based black-box optimization. *arXiv preprint arXiv:2501.19161*, 2025. URL https://arxiv.org/abs/2501.19161.

Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017. doi: 10.1007/s10208-015-9296-2.

Du Phan, Matthew Douglas Hoffman, David Dohan, Sholto Douglas, Tuan Anh Le, Aaron Parisi, Pavel Sountsov, Charles A. Sutton, Sharad Vikram, and Rif A. Saurous. Training chain-of-thought via latent-variable inference. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-3184. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/e69a9560c450ca76584d9eb37e7f5ae8-Abstract-Conference.html.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. COLD decoding: Energy-based constrained text generation with langevin dynamics. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/3e25d1aff47964c8409fd5c8dc0438d7-Abstract-Conference.html.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5377–5400, 2024. doi: 10.18653/v1/2024.naacl-long.301.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, 2025. doi: 10.18653/v1/2025.emnlp-main.36.

Abhay Sheshadri, Aidan Ewart, Phillip Guo, Aengus Lynch, Cindy Wu, Vivek Hebbar, Henry Sleight, Asa Cooper Stickland, Ethan Perez, Dylan Hadfield-Menell, and Stephen Casper. Latent adversarial training improves robustness to persistent harmful behaviors in LLMs. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=6LxMeRlkWl.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, and Sam Toyer. A StrongREJECT for empty jailbreaks. In *Advances in Neural Information Processing Systems*, volume 37, pp. 125416–125440, 2024. doi: 10.52202/079017-3984. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e2e06adf560b0706d3b1ddfca9f29756-Abstract-Datasets_and_Benchmarks_Track.html.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.824. URL https://aclanthology.org/2023.findings-acl.824/.

Peihao Wang, Ruisi Cai, Zhen Wang, Hongyuan Mei, Qiang Liu, Pan Li, and Zhangyang Wang. ∇-Reasoner: LLM reasoning via test-time gradient descent in latent space. In *International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/4b8001fc75f0532827472ea5a16af9ca-Abstract-Conference.html.

Siyuan Wang, Yanchen Liu, and Xiang Ren. Segment-level attribution for selective learning of long reasoning traces. In *International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/38d9676061e0601823e6227d4979e962-Abstract-Conference.html.

Zijun Wang, Haoqin Tu, Yuhan Wang, Juncheng Wu, Yanqing Liu, Jieru Mei, Brian R. Bartoldson, Bhavya Kailkhura, and Cihang Xie. STAR-1: Safer alignment of reasoning LLMs with 1k data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 37988–37997, 2026c. doi: 10.1609/aaai.v40i44.41136. URL https://ojs.aaai.org/index.php/AAAI/article/view/41136.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a00548031e4647b13042c97c922fadf1-Abstract-Conference.html.

Nevan Wichers, Carson Denison, and Ahmad Beirami. Gradient-based language model red teaming. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2862–2881, 2024. doi: 10.18653/v1/2024.eacl-long.175.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696.

Sophie Xhonneux, Alessandro Sordoni, Stephan Günnemann, Gauthier Gidel, and Leo Schwinn. Efficient adversarial training in LLMs with continuous attacks. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/0302fb83c62991efbccf0a003e4f5a92-Abstract-Conference.html.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132, 2024. URL https://proceedings.mlr.press/v235/xia24c.html.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Jiarui Yao, Yifan Hao, Hanning Zhang, Hanze Dong, Wei Xiong, Nan Jiang, and Tong Zhang. Optimizing chain-of-thought reasoners via gradient variance minimization in rejection sampling and RL. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5450.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. LIMO: Less is more for reasoning. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=T2TZ0RY4Zk.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. AdaptThink: Reasoning models can learn when to think. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3716–3730, 2025. doi: 10.18653/v1/2025.emnlp-main.184. URL https://aclanthology.org/2025.emnlp-main.184/.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. FreeLB: Enhanced adversarial training for natural language understanding. In *International Conference on Learning Representations*, 2020.

A Gradient Estimation and Proxy Computation

A.1 Gradients through Explicit Reasoning

For the task loss $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$ in Section 3.1, the reasoning trace $\boldsymbol{z}$ depends on the continuous variable $\boldsymbol{u}$. Discrete token selection prevents ordinary backpropagation from capturing changes in the generated trace. Three common approaches handle this dependence differently.

Fixed-trace differentiation. Holding $\boldsymbol{z}$ fixed gives $\boldsymbol{g}_{\text{fixed}} = \nabla_{\boldsymbol{u}} \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})$, as in GReaTer (Das et al., 2025). The model recomputes hidden states while processing the fixed tokens, capturing how $\boldsymbol{u}$ affects the answer through this computation. Changes in the selected reasoning tokens are excluded.

Continuous relaxations and straight-through estimation. A soft distribution $\mathrm{softmax}(\boldsymbol{o}_t/\tau)$ over next-token logits $\boldsymbol{o}_t$, with temperature $\tau > 0$, produces a weighted token embedding for the next step, making the relaxed computation differentiable (Goyal et al., 2017). A straight-through estimator keeps discrete tokens in the forward pass and substitutes a differentiable backward rule (Bengio et al., 2013; Wang et al., 2026a). Both approaches obtain a direction through an approximation to token selection.

Sampling-based estimation. For stochastic reasoning, let $J(\boldsymbol{u}) = \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{u}, x)}[\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})]$. When differentiation and expectation can be interchanged,

$$\nabla_{\boldsymbol{u}} J(\boldsymbol{u}) = \mathbb{E}_{\boldsymbol{z}}[\nabla_{\boldsymbol{u}} \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) + \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) \nabla_{\boldsymbol{u}} \log p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{u}, x)] \,. \tag{2}$$

The first term differentiates the loss with the sampled trace fixed; the second accounts for changes in its probability (Williams, 1992). Averaging sampled contributions estimates this gradient without differentiating through discrete sampling (Phan et al., 2023; Yao et al., 2025).

A.2 Proxy Gradients in the Original Variable Space

A shorter proxy computation retains the downstream variable $\boldsymbol{u}$. Its gradient must therefore use the coordinates expected by the original algorithm: reasoning-token positions for selective SFT, question-embedding coordinates for safety alignment, vocabulary coefficients for prompt optimization, and response logits for test-time reasoning.

For each gradient query, let ψ_r map $\boldsymbol{u}$ into proxy route r and let ℓ_r be its scalar loss. Backpropagation through this mapping gives

$$\widehat{\boldsymbol{g}}_r = \nabla_{\boldsymbol{u}} \ell_r(\psi_r(\boldsymbol{u})) = J_{\psi_r}(\boldsymbol{u})^\top \nabla_{\psi_r(\boldsymbol{u})} \ell_r. \tag{3}$$

Thus, a shorter internal trace still produces a gradient indexed by the original variable. For selective SFT, differentiating the scalar answer loss returns a gradient tensor over the original token embeddings, whose components provide the token attributions (Appendix D.1). For example, a token-coefficient vector can be mapped through each model's embedding matrix while its gradient remains indexed by the same vocabulary. In test-time reasoning, proxy likelihood gradients and the original reward gradient are both expressed in response-logit coordinates and combined for the update (Appendix D.4).

GP-DA omits the intermediate trace and shares the target parameters. GP-LR differentiates through its continuous latent computation; freezing its parameters preserves the dependence on $\boldsymbol{u}$. GP-Hybrid combines the gradients in these common coordinates as $\widehat{\boldsymbol{g}}_{\text{HY}} = \lambda \widehat{\boldsymbol{g}}_{\text{DA}} + (1-\lambda)\widehat{\boldsymbol{g}}_{\text{LR}}$. Any normalization required by the downstream algorithm is applied after this combination.

B High-Query Reference

Variable and loss evaluation. The variable $\boldsymbol{u} \in \mathbb{R}^{T \times d}$ contains the embeddings of the T question tokens, with embedding dimension d; the model parameters and other prompt tokens remain fixed. For each perturbed input $\boldsymbol{u}'$, the target greedily generates a trace $\boldsymbol{z}(\boldsymbol{u}')$ and evaluates the mean token negative log-likelihood of the specified target answer. This defines $F(\boldsymbol{u}') = \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}', \boldsymbol{z}(\boldsymbol{u}'))$, including the effect of the perturbation on the generated reasoning.

Reference estimation. Flattening $\boldsymbol{u}$ gives $m = Td$ coordinates. The Gaussian-smoothed objective is

$$F_\sigma(\boldsymbol{u}) = \mathbb{E}_{\boldsymbol{v} \sim \mathcal{N}(0, \mathbf{I}_m)}[F(\boldsymbol{u} + \sigma \boldsymbol{v})] \,. \tag{4}$$

The paired estimator in Eq. 1 estimates $\nabla_{\boldsymbol{u}} F_\sigma(\boldsymbol{u})$ (Nesterov & Spokoiny, 2017). We use $K = 1024$ pairs, requiring $2K = 2048$ evaluations, each including reasoning generation and answer scoring. Table 1 measures directional agreement by cosine similarity to this estimate. Figure 4 shows agreement with a $K = 2048$ estimate as the query budget increases.

Target-model update. The probe scale σ determines the smoothing used for reference estimation; the update radius R determines the subsequent perturbation applied to the target. For each nonzero gradient estimate, we use $\delta = -R\widehat{\boldsymbol{g}}/\|\widehat{\boldsymbol{g}}\|_2$, with the norm taken over all question-embedding coordinates. All estimators use the same R for each example. The target regenerates its trace at $\boldsymbol{u} + \delta$, and $\Delta J = F(\boldsymbol{u}) - F(\boldsymbol{u} + \delta)$ measures the decrease in its unsmoothed answer loss. Thus, cosine similarity evaluates agreement with the reference direction, while ΔJ evaluates the resulting target-model update.

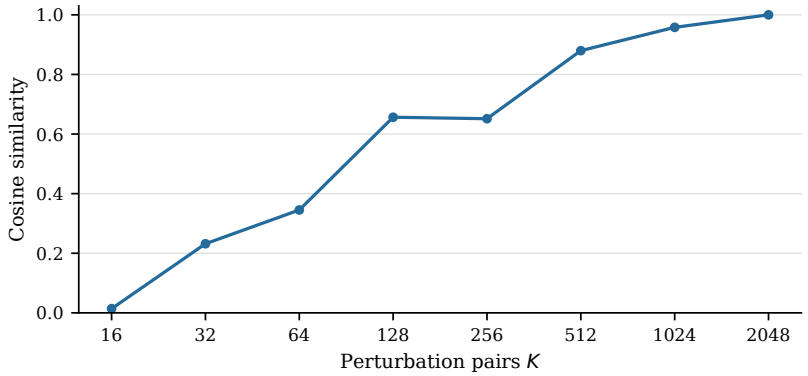


Figure 4: Query-budget calibration: cosine similarity to the estimate with $K = 2048$ perturbation pairs.

C Gradient Comparison: Estimators and Evaluation

All estimators return a direction in the question-token embedding coordinates used by the target (Appendix A.2). Model parameters remain fixed. We compare the following gradient computations in Table 1.

Teacher-Forcing and STE baselines. Teacher-Forcing differentiates the answer loss while holding the generated reasoning trace fixed. STE retains discrete reasoning tokens in the forward pass and uses a surrogate backward rule to propagate gradients to the question embeddings.

Proxy computations. GP-DA uses the fine-tuned target's non-thinking mode and shares its parameters. GP-LR uses $L' = 6$ latent steps and differentiates through the latent computation. GP-Hybrid combines its direction with GP-DA using $\lambda = 0.5$. The resulting question-token gradients are applied at the corresponding target positions, following the coordinate convention in Appendix A.2.

Evaluation. We measure cosine similarity between each nonzero estimate and the high-query reference. To evaluate optimization utility, each direction is normalized to the same perturbation radius R and applied to the target. The target regenerates its reasoning before answer scoring, and the decrease in loss gives ΔJ. This separates agreement with the reference from the effectiveness of an update on the original model. Runtime measures the computation of the gradient estimate.

D Downstream Experimental Details

The algorithms below follow the downstream methods, with boxed gradient queries marking the computation replaced by GP-DA, GP-LR, or GP-Hybrid. All gradients are expressed in the original variable $\boldsymbol{u}$ as described in Appendix A.2. Unless specified otherwise, GP-LR uses $L' = 6$ and GP-Hybrid uses $\widehat{\boldsymbol{g}}_{\text{HY}} = \lambda\widehat{\boldsymbol{g}}_{\text{DA}} + (1-\lambda)\widehat{\boldsymbol{g}}_{\text{LR}}$ with $\lambda = 0.5$.

Shared target initialization. Within each model–task comparison, baseline and proxy branches start from identical target checkpoints after fine-tuning to enable direct answering. GP-DA shares the target parameters; GP-LR uses separately distilled parameters that remain frozen during downstream optimization.

D.1 Data Preparation: Selective Supervised Fine-Tuning

Data and objective. We use 817 LIMO examples for segment-selective SFT (Wang et al., 2026b). Each example provides an annotated CoT trace, which remains fixed during attribution. Its token embeddings $\boldsymbol{u}$ and correct answer a^* define the answer loss $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) = -\log p_{\boldsymbol{\theta}}(a^* \mid \boldsymbol{u}, \boldsymbol{z})$. Model parameters remain frozen during attribution; the selected segments subsequently determine the SFT loss mask.

Gradient replacement. GP-DA and GP-LR provide answer-loss gradients with respect to the annotated trace embeddings. We substitute these gradients into the original integrated-gradient attribution procedure. GP-Hybrid combines the two proxy gradients before attribution; segment scoring and supervision selection follow the original method.

From gradients to segment scores. Following Wang et al. (2026b), we use integrated gradients to measure how reasoning tokens contribute to the correct answer. Let $\boldsymbol{u}_0$ contain padding-token embeddings and let $\boldsymbol{u}^{(j)} = \boldsymbol{u}_0 + t_j(\boldsymbol{u} - \boldsymbol{u}_0)$ for J equally spaced points $t_j \in [0, 1]$. If $\boldsymbol{g}^{(j)}$ denotes the chosen loss gradient at the j-th point, token i receives attribution

$$\alpha_i = -\left(\boldsymbol{u}_i - \boldsymbol{u}_{0,i}\right)^\top \left(\frac{1}{J}\sum_{j=1}^{J} \boldsymbol{g}_i^{(j)}\right). \tag{5}$$

The minus sign converts the loss gradient into attribution to answer log-likelihood. For proxy $r \in \{\mathrm{DA}, \mathrm{LR}\}$, we use

$$\widehat{\boldsymbol{g}}_r^{(j)} = \nabla_{\boldsymbol{u}} \mathcal{L}_{\widehat{\boldsymbol{\theta}}_r}(\boldsymbol{u}, \widehat{\boldsymbol{z}}_r)\Big|_{\boldsymbol{u}=\boldsymbol{u}^{(j)}}. \tag{6}$$

GP-Hybrid interpolates these gradients at each point before computing α_i.

For a segment S_k, we compute attribution strength s_k and direction consistency c_k:

$$s_k = \frac{\sum_{i\in S_k} |\alpha_i|}{\sqrt{|S_k|}}, \qquad c_k = \frac{\left|\sum_{i\in S_k} \alpha_i\right|}{\sum_{i\in S_k} |\alpha_i|}. \tag{7}$$

Strength measures total influence with length normalization; consistency measures whether token contributions have the same sign. We rank segments by s_k and take the smallest prefix accounting for at least 70% of total strength, then retain segments with $c_k \le 0.8$. This selects influential segments containing both positive and negative contributions. The first and final two response segments are also retained. Tokens in selected and boundary segments receive $m_i = 1$, and other response tokens receive $m_i = 0$, giving the masked SFT loss

$$\ell_{\mathrm{SFT}}(\boldsymbol{\theta}) = -\frac{\sum_i m_i \log p_{\boldsymbol{\theta}}(y_i \mid x, y_{<i})}{\sum_i m_i}, \qquad \boldsymbol{y} = (\boldsymbol{z}, a^*). \tag{8}$$

The complete response remains in the conditioning context, and SFT updates the target using its own parameter gradients.

Training settings. We use $J = 50$ interpolation points. SFT runs for 10 epochs with learning rate 3×10^{-5}, batch size 2, and LoRA rank 16 with scaling 32. The R1 CODI proxy is distilled for 3 epochs with learning rate 8×10^{-4} and effective batch size 128.

Evaluation and cost. We use greedy decoding with the benchmark prompts and graders. In-domain accuracy averages MATH-500, AMC23, and AIME24; out-of-domain accuracy averages GPQA-Diamond, Minerva, and OlympiadBench, with equal dataset weights. Time Cost reports the forward and backward computation for attribution, normalized to Selective SFT (100%); GP-Hybrid includes both proxy routes.

Algorithm 1 Segment-selective SFT with a replaceable gradient query

Input: Annotated examples $(x, \boldsymbol{z}, a^*)$, target $p_{\boldsymbol{\theta}}$, J interpolation points.

For each example, use its annotated CoT trace to form embeddings $\boldsymbol{u}$, baseline $\boldsymbol{u}_0$, and segment spans $\{S_k\}$.

  For $j = 1, \ldots, J$, set $\boldsymbol{u}^{(j)} = \boldsymbol{u}_0 + t_j(\boldsymbol{u} - \boldsymbol{u}_0)$.

    $\boxed{\boldsymbol{g}^{(j)} \leftarrow \nabla_{\boldsymbol{u}} \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z})|_{\boldsymbol{u}=\boldsymbol{u}^{(j)}}}$ **Replaceable query:** use a proxy gradient from Eq. 6, optionally combined by Hybrid.

  Attribute each token: $\alpha_i \leftarrow -(\boldsymbol{u}_i - \boldsymbol{u}_{0,i})^\top \big(J^{-1} \sum_j \boldsymbol{g}_i^{(j)}\big)$.

  Compute $s_k \leftarrow \sum_{i \in S_k} |\alpha_i| / \sqrt{|S_k|}$ and $c_k \leftarrow |\sum_{i \in S_k} \alpha_i| / \sum_{i \in S_k} |\alpha_i|$; rank by s_k.

  Take the smallest prefix with cumulative strength $\geq 0.7 \sum_k s_k$; retain its segments with $c_k \leq 0.8$.

  Add the first and final two response segments; set $m_i = 1$ on retained tokens and 0 elsewhere.

Fine-tune $p_{\boldsymbol{\theta}}$ with the masked loss in Eq. 8.

Output: The selectively fine-tuned target model.

D.2 Training: Safety Alignment

Data and training objective. We use the STAR-1 mixture of 1,000 harmful and 915 benign instructions, each paired with a supervised response (Wang et al., 2026c). For harmful instructions, the inner objective is $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) = \ell_{\text{unsafe}} - \ell_{\text{safe}}$, where $\boldsymbol{u} = \delta$ perturbs the question embeddings. Minimizing it favors the unsafe answer over the safe reference. For benign instructions, the undesirable target is a refusal, so the analogous contrast encourages over-refusal. In both cases, the outer update learns the correct reference response from equally weighted clean and perturbed inputs.

Gradient replacement. Teacher Forcing AT computes the input gradient while conditioning on a fixed explicit reasoning trace. GP-DA supplies a proxy gradient from the current target's direct-answer computation, whereas GP-LR uses a frozen CODI proxy. GP-Hybrid interpolates their gradients with weight λ. The resulting gradient is normalized for each update to $\boldsymbol{u}$ within an ℓ_2 budget. Once the perturbed input has been constructed, its perturbation is held fixed during the outer parameter update. Proxy gradients therefore replace the inner input-gradient computation; the original target still computes its supervised training loss and parameter gradients.

Algorithm 2 Safety adversarial training with a replaceable gradient query

Input: Training examples, target $p_{\boldsymbol{\theta}}$, radius ϵ, step size η, inner steps K.

For each epoch and training batch, obtain the reference and undesirable answers and initialize $\boldsymbol{u}^{(0)}$ inside the ℓ_2 ball.

  For $k = 0, \ldots, K-1$:

    $\boxed{\boldsymbol{g}^{(k)} \leftarrow \nabla_{\boldsymbol{u}} (\ell_{\text{unsafe}} - \ell_{\text{safe}})|_{\boldsymbol{u}=\boldsymbol{u}^{(k)}}}$ **Replaceable query.**

    $\boldsymbol{u}^{(k+1)} \leftarrow \Pi_{\|\cdot\|_2 \leq \epsilon}\big(\boldsymbol{u}^{(k)} - \eta \boldsymbol{g}^{(k)} / \|\boldsymbol{g}^{(k)}\|_2\big)$.

  Hold $\boldsymbol{u}^{(K)}$ fixed; compute reference-response losses on clean and perturbed inputs.

  Update $\boldsymbol{\theta}$ to minimize $\frac{1}{2}\ell_{\text{clean}}(\boldsymbol{\theta}) + \frac{1}{2}\ell_{\text{perturbed}}(\boldsymbol{\theta}; \boldsymbol{u}^{(K)})$.

Output: The safety-aligned target model.

Training settings. We use two inner updates and compute the contrastive loss on the first 12 answer tokens. For T perturbed tokens with mean embedding norm $\bar{e}$, the radius and step size are $\epsilon = 0.05\bar{e}\sqrt{T}$ and $\eta = 0.025\bar{e}\sqrt{T}$. R1 is evaluated over six AT epochs. Llama branches share a three-epoch safety-SFT checkpoint and undergo two further SFT or AT epochs with learning rate 10^{-5} and effective batch size 128. Its CODI proxy is distilled for two epochs at learning rate 8×10^{-4}.

Evaluation and cost. We report the mean StrongREJECT score on complete responses (Souly et al., 2024), and over-refusal on the 250 benign XSTest prompts (Röttger et al., 2024), assigning weights one and one half to full and partial refusals. AutoRAN ASR@7 measures the proportion of prompts with a harmfulness score of at least seven (Liang et al., 2026). Figure 3a shows the epoch trajectory. Reasoning evaluation uses all 500 MATH-500 questions, all 198 GPQA-Diamond questions for both Llama and R1. Time Cost reports the forward and backward computation used to construct input perturbations, normalized to Teacher Forcing AT (100%).

D.2.1 Mixing Weight Analysis

Experimental setup. We vary $\lambda \in \{0, 0.25, 0.5, 0.75, 1\}$ for safety AT on R1, keeping the training and evaluation protocol fixed. The weight λ controls DA guidance, so $\lambda = 0$ and 1 recover pure LR and DA, respectively. We evaluate the resulting models using AutoRAN ASR@7.

Results. All three intermediate weights outperform both endpoints (Table 6). ASR ranges from 58.6% to 59.1%, compared with 62.3% for LR and 61.7% for DA. The best weight, $\lambda = 0.25$, improves on the equal-weight default by 0.5 percentage points. Thus, combining the two signals matters more here than precisely tuning their relative weight: the default $\lambda = 0.5$ retains most of the observed improvement.

Table 6: Effect of the mixing weight on safety AT with R1-Distill-Qwen-1.5B. Lower AutoRAN ASR@7 is better.

λ	0 (GP-LR)	0.25	0.50	0.75	1 (GP-DA)
ASR@7 (%)	62.3	**58.6**	59.1	59.0	61.7

D.3 Task Adaptation: Prompt Optimization

Tasks and objective. We optimize a shared prompt on GSM8K and each of five BBH tasks: object counting, tracking shuffled objects, hyperbaton, causal judgement, and movie recommendation. Following GReaTer (Das et al., 2025), the objective is $\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) = \ell_{\text{answer}} + \gamma \ell_{\text{fluency}}$, where $\boldsymbol{u}$ contains token-selection coefficients at editable prompt positions. Model parameters remain fixed; the selected prompt is reused on held-out questions.

Gradient replacement and search. At each editable position, the target proposes likely tokens. We average answer-loss gradients over the optimization examples and combine them with the target's fluency gradient. For current token p_i, the first-order loss change for replacing it with v is $\Delta_{i,v} \approx g_{i,v} - g_{i,p_i}$, where $\boldsymbol{g} = \boldsymbol{g}_{\text{answer}} + \gamma \nabla_{\boldsymbol{u}} \ell_{\text{fluency}}$. Replacements with the lowest estimated changes form a shortlist. The target generates answers with each shortlisted prompt, and GReaTer selects the edit by answer error with a fluency penalty. GP-DA and GP-LR replace $\boldsymbol{g}_{\text{answer}}$ using direct answering or latent reasoning; GP-Hybrid mixes these two gradients before ranking edits. Candidate generation, fluency computation, and evaluation continue to use the target model.

Algorithm 3 GReaTer prompt search with a replaceable gradient query

Input: Optimization examples $\mathcal{D}$, initial prompt p, frozen target $p_{\boldsymbol{\theta}}$.
For each search step, choose the next editable position i.
  Form the candidate set $\mathcal{V}_i$ from the target's likely next tokens.
  $\boxed{\boldsymbol{g}_{\text{answer}} \leftarrow \nabla_{\boldsymbol{u}} \ell_{\text{answer}}}$ **Replaceable query.**
  $\boldsymbol{g} \leftarrow \boldsymbol{g}_{\text{answer}} + \gamma \nabla_{\boldsymbol{u}} \ell_{\text{fluency}}$.
  For $v \in \mathcal{V}_i$, rank edits by $\Delta_{i,v} = g_{i,v} - g_{i,p_i}$ and shortlist the lowest-scoring edits.
  Generate target-model answers for each shortlisted prompt $p^{i \to v}$ on $\mathcal{D}$.
  Choose the edit with the lowest answer-error and fluency selection score; update p_i.
Output: The selected prompt, reused on held-out questions.

Search settings. We use 50 optimization examples per task and 105 search steps. Each step starts from the target's top-10 token candidates and shortlists 3 edits for evaluation. The fluency weight and answer-logit temperature are both 0.2. Target and proxy parameters remain fixed during search.

Evaluation and cost. The unoptimized CoT prompt is the initial-prompt control. We report accuracy on GSM8K and each BBH task, with Avg. denoting the unweighted mean across all six tasks. Time Cost reports the forward and backward computation of edit directions, normalized to GReaTer (100%) on the same model.

D.4 Inference: Test-Time Reasoning

Tasks and objective. We evaluate on 800 GSM8K questions, an 83-question AMC set, and all 500 MATH-500 questions. Following ∇-Reasoner (Wang et al., 2026a), the target generates an

Algorithm 4 ∇-Reasoner decoding with a replaceable likelihood-gradient query

Input: Question x, frozen target, reward model $\mathcal{R}$, weights α, β, search budget.
1 **Until** the response is complete, generate a continuation y from the current prefix and initialize logits $\boldsymbol{u}$.
2 **For each** DTO step, form the straight-through tokens in Eq. 9.
3 $\boxed{\boldsymbol{g}_{\mathrm{NLL}} \leftarrow \nabla_{\boldsymbol{u}} \ell_{\mathrm{NLL},\boldsymbol{\theta}}(\boldsymbol{u})}$ **Replaceable query.**
4 $\boldsymbol{g} \leftarrow \alpha \boldsymbol{g}_{\mathrm{NLL}} + \beta \nabla_{\boldsymbol{u}} \ell_{\mathrm{reward}}$; update $\boldsymbol{u}$ with Adam using $\boldsymbol{g}$.
5 Propose token v from the refined first-position logits.
6 If v differs from the original next token, generate continuation y' beginning with v and accept it only if $\mathcal{R}(y'; x) > \mathcal{R}(y; x)$; otherwise keep the original token.
7 Append the accepted token to the prefix and continue under the same search budget.
Output: The completed response.

initial response and its token logits. We optimize these logits, denoted $\boldsymbol{u} \in \mathbb{R}^{T \times |\mathcal{V}|}$, with target and reward-model parameters fixed. At response position i, the straight-through token representation is

$$\boldsymbol{y}_i(\boldsymbol{u}) = \mathrm{onehot}(\arg\max_j u_{ij}) + \mathrm{softmax}(\boldsymbol{u}_i/\tau) - \mathrm{sg}[\mathrm{softmax}(\boldsymbol{u}_i/\tau)]\,, \tag{9}$$

where τ is the temperature and sg stops gradients. The language model and reward model receive these token representations through their respective embedding matrices. In particular, the reward input is $\boldsymbol{y}(\boldsymbol{u})E_{\mathrm{RM}}$, conditioned on the question and accepted prefix using the original chat template; E_{RM} is aligned to the language-model vocabulary.

The objective combines sequence likelihood and reward:

$$\begin{aligned} \ell_{\mathrm{NLL},\boldsymbol{\theta}}(\boldsymbol{u}) &= -\sum_{i=1}^{T} \boldsymbol{y}_i(\boldsymbol{u})^\top \log p_{\boldsymbol{\theta}}(\cdot \mid x, \boldsymbol{y}_{<i}(\boldsymbol{u})), \\ \ell_{\mathrm{reward}}(\boldsymbol{u}) &= -\mathcal{R}(\boldsymbol{y}(\boldsymbol{u}); x), \\ \mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{u}, \boldsymbol{z}) &= \alpha \ell_{\mathrm{NLL},\boldsymbol{\theta}}(\boldsymbol{u}) + \beta \ell_{\mathrm{reward}}(\boldsymbol{u}). \end{aligned} \tag{10}$$

Here, α and β weight likelihood and reward, and $\boldsymbol{z}$ is the reasoning portion of the response. The accepted prefix is included in the fixed conditioning context. Both losses differentiate with respect to the same response logits $\boldsymbol{u}$.

Gradient replacement. GP-DA and GP-LR replace the likelihood gradient while retaining the original reward gradient. Let $\widehat{\ell}_{\mathrm{NLL},r}$ denote the proxy likelihood loss for $r \in \{\mathrm{DA}, \mathrm{LR}\}$. The update direction is

$$\widehat{\boldsymbol{g}}_r = \alpha \nabla_{\boldsymbol{u}} \widehat{\ell}_{\mathrm{NLL},r}(\boldsymbol{u}) + \beta \nabla_{\boldsymbol{u}} \ell_{\mathrm{reward}}(\boldsymbol{u}). \tag{11}$$

GP-Hybrid interpolates the DA and LR likelihood gradients before adding the reward term. Algorithm 4 uses this direction to refine response logits, while candidate generation and reward-based acceptance follow ∇-Reasoner.

Search settings. Response logits are optimized with Adam at learning rate 0.01 for at most 100 iterations, with reward coefficient $\beta = 1$. We allow up to 8 generated candidates and a maximum sequence length of 3,072 tokens. All gradient routes use the same decoding and acceptance settings.

Evaluation and cost. We extract and grade the selected answers with the task evaluators. Greedy decoding is evaluated for accuracy only. Time Cost measures response search, normalized to ∇-Reasoner (100%) on each dataset and averaged equally across the three datasets. The latent-depth ablation reports the cost of computing the update direction per question.