VCORE: Variance-Controlled Optimization-based Reweighting for Chain-of-Thought Supervision

Xuan Gong[1], **Senmiao Wang**[2], **Hanbo Huang**[1], **Ruoyu Sun**[2], **Shiyu Liang**[1*]
[1] Shanghai Jiao Tong University, [2] Chinese University of Hong Kong (Shenzhen),
{gongxuan0610, hhuang417, lsy18602808513}@sjtu.edu.cn,
senmiaowang1@link.cuhk.edu.cn, ruoyus@illinois.edu

Abstract

Supervised fine-tuning (SFT) on long chain-of-thought (CoT) trajectories has emerged as a crucial technique for enhancing the reasoning abilities of large language models (LLMs). However, the standard cross-entropy loss treats all tokens equally, ignoring their heterogeneous contributions across a reasoning trajectory. This uniform treatment leads to misallocated supervision and weak generalization, especially in complex, long-form reasoning tasks. To address this, we introduce **V**ariance-**C**ontrolled **O**ptimization-based **RE**weighting (VCORE), a principled framework that reformulates CoT supervision as a constrained optimization problem. By adopting an optimization-theoretic perspective, VCORE enables a principled and adaptive allocation of supervision across tokens, thereby aligning the training objective more closely with the goal of robust reasoning generalization. Empirical evaluations demonstrate that VCORE consistently outperforms existing token reweighting methods. Across both in-domain and out-of-domain settings, VCORE achieves substantial performance gains on mathematical and coding benchmarks, using models from the Qwen3 series (4B, 8B, 32B) and LLaMA-3.1-8B-Instruct. Moreover, we show that VCORE serves as a more effective initialization for subsequent reinforcement learning, establishing a stronger foundation for advancing the reasoning capabilities of LLMs.

1 Introduction

Recent advances in LLMs have highlighted the impressive benefit of long chain-of-thought (long CoT) for enhancing the reasoning capabilities. Recent LLMs, such as OpenAI-o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025a), Kimi k1.5 (Team et al., 2025a), and Qwen3 series (Yang et al., 2025), pursue this direction by scaling CoT lengths and show strong reasoning performance on tasks such as challenging math problem solving and code generation benchmarks.

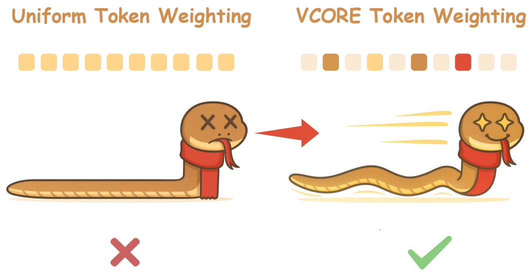

Figure 1: **Overview of VCORE.** Compared to the standard cross-entropy loss, VCORE approaches long-CoT SFT from an optimization perspective and adjusts token weights according to their gradient utility, thereby enabling more effective use of supervision signals and improving generalization.

Beyond reinforcement learning (RL) and test-time methods, long chain-of-thought (CoT) supervised fine-tuning (SFT) has been increasingly adopted by AI labs and companies (Team, 2025; Muennighoff et al., 2025; Xu et al., 2025; Labs, 2025; Ye et al., 2025). Compared with these two paradigms, long-CoT SFT typically distills reasoning traces from teacher models or curated datasets, offering a more straightforward yet effective route to improving reasoning. Nevertheless, most existing long-CoT SFT works emphasize engineering implementations and data recipes, leaving substantial blanks on the *optimization-algorithm progress*.

Unfortunately, long-CoT SFT is particularly susceptible to supervision noise, which can lead to misallocated supervision and degraded generalization (Luo et al., 2025; Lobo et al., 2025). To a large extent, this susceptibility can be attributed to the convention of uniform token weighting in the cross-entropy loss across lengthy reasoning traces (e.g. exceeding 1k tokens). A growing body of evidence shows that: (1) not all intermediate tokens are equally worth learning from (Choi et al., 2025; Li et al., 2025); (2) spurious or unfaithful tokens

[*]Corresponding author

Preprint. Under Review.

may corrupt learning signals (Chen et al., 2025b; Turpin et al., 2023; Zhou et al., 2024). These findings motivate *breaking the limitation of uniform token weighting* in long-CoT supervision.

The above discussion leads to a central question:

Can we design a long CoT supervision algorithm that reweights tokens more effectively through an optimization-based approach?

In this paper, we answer this question by proposing **VCORE**: a variance-controlled, optimization-based reweighting framework for CoT supervision. VCORE departs from prior heuristics and formulates token reweighting as a constrained optimization problem: for each trajectory, compute the distribution over token positions that maximizes expected loss descent under a single SGD step, subject to a KL constraint for stability. This yields a closed-form *Gibbs distribution* over token-wise gradient utilities, directly grounded in first-order descent dynamics. To efficiently estimate utilities, VCORE introduces a lightweight *one-backward probing trick* that requires only one backward pass and forward-mode perturbations per batch. To ensure stable updates, it further rescales the weights using a principled variance control coefficient α, which matches the update variance to that of uniform weighting. This end-to-end formulation identifies high-impact tokens from the training signal itself, *without relying on teacher guidance, confidence thresholds or entropy filters*. Our contributions are summarized as follows:

- We cast CoT supervision as a constrained optimization problem over token weights, grounded in the dynamics of SGD. This formulation yields a closed-form Gibbs distribution that allocates weight based on token-level gradient utility.
- We propose **VCORE**, a unified framework that combines optimization-derived token reweighting with variance-controlled scaling in a single efficient pipeline.
- VCORE consistently outperforms baseline DFT and iw-SFT on math and code benchmarks, improving both in-domain accuracy and out-of-domain generalization.
- VCORE further strengthens RL fine-tuning by providing a more effective initialization, yielding higher post-RL performance.

2 Related Work

Due to space constraints, we present a concise overview here; the complete related work is provided in Appendix A.

SFT for Reasoning. Beyond chain-of-thought (CoT) prompting (Wei et al., 2022), supervised fine-tuning (SFT) on *long* CoT traces has emerged as a simple and effective way to imbue LLMs with slow, multi-step reasoning, often via distillation from stronger teachers or curated rationales (Team, 2025; Muennighoff et al., 2025; Xu et al., 2025; Labs, 2025). Compared to alternatives that require on-policy rollouts, long-CoT SFT is operationally lightweight, attains strong generalization, and frequently serves as an initialization for subsequent RL-based improvement (Yeo et al., 2025).

Token Reweighting in SFT. Some recent studies modify SFT by adopting token-level loss reweighting. Wu et al. (2025) reinterprets standard SFT as a policy-gradient-style update with an implicit $1/\pi_\theta$ importance factor that over-weights low-probability tokens; it therefore introduces Dynamic Fine-Tuning (DFT), which rectifies the update by multiplying per-token loss by the model's probability for the target token. Qin and Springenberg (2025) shows that SFT on curated/filtered data optimizes a lower bound to an RL objective; accordingly, it proposes Importance-weighted SFT (iw-SFT) which tightens that bound by importance-weighting the SFT log-likelihood relative to a reference or current policy.

Our method (VCORE) is also an SFT-stage token-level reweighting, yet it is *conceptually distinct* from DFT and iw-SFT. Whereas DFT and iw-SFT are *RL-motivated*, VCORE is *optimization-driven*: we formulate the choice of token weights as maximizing the first-order loss decrease of a single SGD step subject to a KL constraint. This yields a closed-form Gibbs weighting over tokens. We further introduce an explicit variance-control mechanism that matches the update variance of the reweighted supervision to that under uniform weighting, stabilizing training on long CoT sequences. Empirically, VCORE delivers stronger reasoning generalization over DFT and iw-SFT within the same SFT protocol.

3 Preliminaries

Notations. Let $\mathcal{V}$ be a finite vocabulary, and let $\mathcal{V}^*$ denote the set of all finite-length sequences over $\mathcal{V}$. Each training instance (x, y) consists of an input prompt x and a target sequence $y = (y_1, y_2, \dots) \in \mathcal{V}^*$ of length $|y|$, typically encoding a reasoning trajectory distilled from a teacher model via CoT supervision. The goal is to train a student model p_θ, parameterized by θ, to imitate these trajectories through next-token prediction. The language model p_θ defines a conditional distribution over tokens, with the sequence likelihood factorized autoregressively as $p_\theta(y|x) = \prod_{t=1}^{|y|} p_\theta(y_t|x, y_{<t})$, where $y_{<t} = (y_1, \ldots, y_{t-1})$ denotes the prefix up to position $t-1$. Each term is computed by applying a softmax over the output logits, yielding a distribution over the next token conditioned on the input and the prefix.

Naive CoT Supervision. We train the model p_θ by minimizing the expected per-token log-loss under the true data distribution $\mathcal{P}$, defined as $\mathcal{L}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{P}}\left[\frac{-\log p_\theta(y|x)}{|y|}\right]$. In practice, we optimize this objective over a finite training dataset $\mathcal{D}$ by iteratively sampling mini-batches $\mathcal{B} \subset \mathcal{D}$ and applying gradient-based updates of the form $\theta^+ \leftarrow \theta - \mathcal{T}(\nabla_\theta \hat{\mathcal{L}}_\mathcal{B}(\theta))$, where $\mathcal{T}$ denotes a generic update operator (e.g., SGD or Adam) that acts on the loss gradient and, when applicable, historical statistics. The mini-batch gradient takes the following form:

$$\nabla_\theta \hat{\mathcal{L}}_\mathcal{B}(\theta) = \sum_{(x,y)\in\mathcal{B}} \left[-\frac{1}{|y|}\nabla_\theta \log p_\theta(y \mid x)\right]$$

$$\triangleq \sum_{(x,y)\in\mathcal{B}} \left[\sum_{t\geq 1} \underbrace{(1/|y|)}_{\text{uniform}} \cdot \nabla_\theta \ell_t(\theta; x, y)\right].$$

Here, $\ell_t(\theta; x, y) = -\log p_\theta(y_t \mid x, y_{<t})$ denotes the token-level prediction loss at position t. The resulting gradient update corresponds to a uniform average over all next-token prediction tasks, implicitly treating each token as equally informative and equally reliable. This choice is not arbitrary: under the autoregressive factorization, uniform weighting yields an unbiased estimator of the population loss gradient $\nabla_\theta \mathcal{L}(\theta)$, making it a natural default.

Limitations of Uniform Weighting. (1) *Not all tokens are equally worth learning from.* Uniform weighting spreads supervision evenly across positions, regardless of how much signal each token provides. Many next-token predictions are either trivially easy or hopelessly ambiguous-both yield gradients with little learning value. This misallocation wastes updates and slows convergence. (2) *Spurious tokens corrupt learning signals.* Auto-distilled CoTs often contain hallucinated or misaligned tokens. Uniform weighting treats these as equally important, allowing noise to dominate gradients and impair generalization.

Toward Adaptive Token Weighting. A natural remedy to these limitations is to adapt token-wise supervision to the input and model state. Let $q_t(x, y, \theta)$ denote a distribution over positions $t = 1, \ldots, |y|$, conditioned on the input prompt x, target sequence y and current model parameters θ. Unlike the uniform weighting, adaptive q concentrates learning on salient or uncertain positions while suppressing redundant or noisy ones. The reweighted gradient over a mini-batch $\mathcal{B}$ becomes:

$$\nabla_\theta \hat{\mathcal{L}}_\mathcal{B}(\theta; q) = \sum_{(x,y)\in\mathcal{B}} \left[\sum_{t\geq 1} \underbrace{q_t(x, y)}_{\text{adaptive}} \nabla_\theta \ell_t(\theta; x, y)\right]$$

with the update $\theta^+(q) \leftarrow \theta - \mathcal{T}\left(\nabla_\theta \hat{\mathcal{L}}_\mathcal{B}(\theta; q)\right)$, where $\mathcal{T}$ denotes a generic update operator.

Problem: How Should We Choose q? If only a few tokens meaningfully drive learning, then uniform supervision is fundamentally wasteful. To optimize efficiently, we must ask: where should gradients go? The goal of adaptive weighting is to direct supervision toward the most impactful positions that best accelerate learning under the current model. This requires selecting $q(t \mid x, y, \theta)$ based on local gradient utility, rather than adhering to a fixed prior. To avoid instability from overly sharp focus, we constrain q to remain close to uniform:

$$\min_q \mathcal{L}(\theta^+(q)) \quad \text{s.t.} \quad \mathrm{KL}(q(\cdot \mid x, y, \theta) \parallel u) \leq \delta,$$

where $u(t) = 1/|y|$ and δ bounds deviation. This formulation balances targeted supervision with stability, enabling fine-grained, model- and input-aware token selection.

4 Method

4.1 Optimal Reweighting under SGD

In this subsection, we derive the optimal reweighting distribution $q^*(t \mid x, y, \theta)$ that maximizes the loss decrease after a single-step SGD update. By applying a first-order Taylor approximation to the SFT objective, we fortunately derive a closed-form solution for the optimal token-wise weights, enhancing interpretability and enabling efficient algorithmic implementation.

Step 1: First-order Taylor Expansion. Consider an SGD step with learning rate η, using the reweighted gradient: $\theta^+ = \theta - \eta\nabla_\theta \hat{\mathcal{L}}_\mathcal{B}(\theta; q)$. For small η, the population loss at the updated parameters admits a first-order approximation:

$$\mathcal{L}(\theta^+) - \mathcal{L}(\theta) = -\eta \left\langle \nabla\mathcal{L}(\theta), \nabla\hat{\mathcal{L}}_{\mathcal{B}}(\theta; q)\right\rangle + \mathcal{O}(\eta^2)$$
$$= -\eta \sum_{(x,y)\in\mathcal{B}} \sum_{t\geq 1} \Big[q_t(x,y) \cdot s_t(x,y,\theta) \Big] + \mathcal{O}(\eta^2)$$

where we define the per-token gradient utility

$$s_t(x,y,\theta) \triangleq \langle \nabla_\theta \mathcal{L}(\theta), \nabla_\theta \ell_t(\theta; x, y)\rangle\,. \quad (1)$$

This quantity captures the alignment between the global descent direction and the gradient induced by supervising token y_t. Tokens with higher s_t contribute more to reducing the population loss and should be prioritized.

Step 2: Optimal Adaptive Weighting. Maximizing the descent in the first-order expansion reduces to a constrained optimization over the probability simplex Δ at each training instance (x, y):

$$\max_{q\in\Delta} \sum_{t\geq 1} q(t) \cdot s_t(x,y) \quad \text{s.t.} \quad \mathrm{KL}(q \parallel u) \leq \delta$$
$$\Longrightarrow \quad q^*(t \mid x, y, \theta) = \frac{\exp(\tau s_t(x,y,\theta))}{\sum_{j\geq 1} \exp(\tau s_j(x,y,\theta))},$$

where $\tau > 0$ is a temperature set by the constraint. This standard exponential tilting problem admits a unique *closed-form solution*: a Gibbs distribution over token-level gradient utilities. As $\tau \to 0$, q^* recovers the uniform prior; as $\tau \to \infty$, it concentrates on the highest-utility tokens.

Step 3: A One-Backward Trick for Estimating s_t. Estimating the token utility $s_t(x, y, \theta)$ in Equation (1) naively requires one backward pass per token, computationally infeasible for long sequences. We introduce a non-trivial and highly efficient solution: a *one-backward forward-mode probing trick* that recovers all s_t values using just a single gradient step and $|y|$ forward evaluations. Specifically, we draw an independent mini-batch $\mathcal{B}' \sim \mathcal{P}$, compute the descent direction $\nabla_\theta \mathcal{L}_{\mathcal{B}'}(\theta; u)$ under uniform token weights, and measure the change in the token-wise loss after a small perturbation in that direction. This yields an unbiased estimator:

$$\lim_{\epsilon\to 0} \frac{\mathbb{E}_{\mathcal{B}'}[\ell_t(\theta; x, y) - \ell_t(\theta - \epsilon\nabla_\theta \mathcal{L}_{\mathcal{B}'}(\theta; u); x, y)]}{\epsilon}$$
$$= \langle \nabla_\theta \mathcal{L}(\theta), \nabla_\theta \ell_t(\theta; x, y)\rangle = s_t(x, y, \theta).$$

This construction reduces the cost of estimating all s_t values from $|y|$ backward passes to just **one backward** (to compute the descent direction from $\mathcal{B}'$) and $|y|$ forward passes (to evaluate perturbed token losses). It requires no second-order gradients, no backward hooks and no additional model queries. By leveraging the directional derivative structure of s_t, this probing trick makes adaptive token weighting both scalable and plug-and-play in standard training pipelines.

Beyond Heuristics: Token Weighting as Optimization, Not Guesswork. Some prior works rank tokens by entropy (Wang et al., 2025a) or by their estimated influence on outcome correctness (Lin et al., 2024; Vassoyan et al., 2025), typically without leveraging training-time gradient information. In contrast, we take an optimization perspective and derive a closed-form optimal token-weighting distribution, moving beyond heuristic rules. Crucially, our method identifies the most impactful token positions directly from the training signal—*without relying on teacher guidance or manually tuned thresholds*.

4.2 VCORE: Variance-Controlled Optimization-based REweighting

We introduce **VCORE**, a **V**ariance-**C**ontrolled **O**ptimization-based **RE**weighting framework for chain-of-thought supervision, which is showed in Algorithm 1. Building on Section 4.1, which presented the Gibbs-form reweighting $q^*(t \mid x, y, \theta) \propto \exp(\tau s_t)$ and a one-backward token-utility estimation trick, we now add a variance-normalization mechanism to control the update variance at each training step. Concretely, we rescale the parameter update by an adaptive coefficient α in order to align the variance of the reweighted supervision with that of the uniform supervision. This trick helps stabilize the training dynamics while allowing the learning signal to concentrate on informative reasoning tokens. Moreover, it introduces no extra architectural changes.

Variance-Controlled Descent Scaling. The Gibbs reweighting $q^*(t \mid x, y, \theta) \propto \exp(\tau s_t)$ focuses learning on informative tokens, but it also changes the variability of the update. To make this explicit, we define the variance of the (per-batch) update under reweighted supervision *before* any scaling, $\mathcal{V}_q \triangleq \mathrm{Var}[\sum_t q_t\, s_t]$, and the variance under uniform supervision by, $\mathcal{V}_u \triangleq \mathrm{Var}[\sum_t s_t/|y|]$. We then introduce an adaptive coefficient α to control the update magnitude, choosing α to match the variance of reweighted updates with that of uniform weighting:

$$\alpha^2 \mathcal{V}_q = \mathrm{Var}\left[\alpha \sum_t q_t s_t\right]$$
$$\approx \mathrm{Var}\left[\sum_t (s_t/|y|)\right] = \mathcal{V}_u.$$

This yields a choice: $\alpha = \sqrt{\mathcal{V}_u/\mathcal{V}_q}$. If the scaling coefficient α is too small, gradient steps shrink and training slows down; if α is too large, especially under sharply peaked weights, stochastic gradients become highly variable and convergence degrades.

Intuition. If token utilities s_t are uncorrelated with constant variance σ^2, then $\mathcal{V}_u \approx \sigma^2/|y|$. Under a highly peaked Gibbs weighting (e.g., q_t concentrates on one token), $\mathcal{V}_q^{(0)} \approx \sigma^2$, leading to $\alpha = 1/\sqrt{|y|}$. Thus, when q is sharp or sequences are long, aggressive reweighting would amplify variance and α must shrink to stabilize training; when q is balanced, $\alpha \approx 1$ and the full descent step is recovered without loss of stability.

Summary: A New Perspective on CoT Supervision. *(1) Optimization-Derived Weighting.* Existing approaches typically assign token weights using entropy or model confidence, ignoring the optimization dynamics that govern parameter updates. Our framework instead derives $q^*(t)$ as the unique solution to an optimization problem that maximizes population loss reduction under a single SGD step with a KL constraint. This yields a closed-form Gibbs distribution over token-level gradient utilities, grounding token supervision in first-order descent rather than heuristics. *(2) Variance-Controlled Stabilization.* Although q^* is theoretically optimal for maximizing descent, its practical effectiveness depends on the stability of the updates it induces. High temperatures or heavy-tailed utilities can cause the resulting weights to become sharply concentrated, amplifying gradient variance and destabilizing training. Our VCORE framework addresses this by introducing a principled variance-controlled scaling coefficient α that matches the variance of reweighted updates to that of uniform supervision. This preserves the adaptivity of Gibbs weighting while ensuring stable and efficient learning without heuristic clipping or ad-hoc thresholds.

5 Experiments

In this section, we investigate and answer the following research questions:

Algorithm 1 VCORE Algorithm

Require: Dataset $\mathcal{D} = \{(x, y)\}$, model parameters θ, learning rate η, temperature τ, probing scale ϵ
Ensure: Updated parameters θ^+
for each batch $\mathcal{B}$ in $\mathcal{D}$ **do**
    Draw a random batch $\mathcal{B}'$ from $\mathcal{D}$
    Compute descent direction $\nabla_\theta \mathcal{L}_{\mathcal{B}'}(\theta; u)$ under uniform token weights
    Estimate token t score $s_t(x, y, \theta)$ by $\lim_{\epsilon \to 0} \frac{\mathbb{E}_{\mathcal{B}'}[\ell_t(\theta;x,y) - \ell_t(\theta - \epsilon \nabla_\theta \mathcal{L}_{\mathcal{B}'}(\theta;u);x,y)]}{\epsilon}$
    Get weighs distribution $q^*(t \mid x, y, \theta)$ by $\frac{\exp(\tau s_t(x,y,\theta))}{\sum_j \exp(\tau s_j(x,y,\theta))}$
    Calculate the scaling factor α
    $\theta \leftarrow \theta - \eta\, \mathbb{E}_{(x,y),t\sim q^*}[\nabla_\theta(\alpha \ell_t(\theta; x, y))]$
end for
Get the final trained model parameters $\theta^+ = \theta$

RQ1: Can VCORE achieve better generalization than uniform and heuristic reweighting methods?

RQ2: Which parts of VCORE are essential for improving reasoning and stability across domains?

RQ3: What are the practical implications and limitations of VCORE in CoT training?

5.1 Experimental Setups

Models and Supervised Tasks. We study long CoT supervision on two domains (math and coding), using Qwen3 models (4B, 8B, 32B) (Yang et al., 2025) and LLaMA3.1-8B-Instruct (AI@Meta, 2024). Domain-specific training data are curated from recent high-quality sources: OpenMathReasoning (Moshkov et al., 2025) and the C++ subset of OpenCodeReasoning (NVIDIA, 2025). We retain only CoT instances generated by DeepSeek-R1 (DeepSeek-AI, 2025), applying automatic filtering to ensure trajectory correctness. For Qwen3 models, we sample 3.2k examples per domain and the average CoT length is 3155.01 (math) and 2861.25 (code) ; for LLaMA, we use 32k examples per domain and the average CoT length is 3007.79 (math) and 2805.43 (code). Further preprocessing details and CoT length statistics are provided in Appendix B.1.

Evaluation Benchmarks. We evaluate each model under both *in-domain* and *out-of-domain* generalization settings. **In-domain** benchmarks include the union of AIME 2024 and AIME

Model	Method	Math				Code				Avg.	
		AIME	Olympiad	RBench	SGPQA-1k	LCB	OJBench	RBench	SGPQA-1k	ID	OOD
LLaMA3.1 8B Instruct	Original	3.33	17.80	21.76	18.60	9.67	1.29	21.76	18.60	8.02	20.18
	SFT	3.33	23.59	4.02	12.10	9.29	0.86	5.21	9.50	9.27	7.71
	DFT	3.33	18.84	6.22	13.90	0.00	0.00	3.47	5.10	5.54	7.17
	iw-SFT	5.00	22.55	2.83	11.60	9.00	0.86	3.38	6.60	<u>9.35</u>	6.10
	Random	1.67	22.55	7.95	15.10	6.16	3.02	4.75	8.50	8.35	<u>9.07</u>
	VCORE	6.67	25.96	3.56	12.80	10.33	2.58	6.22	15.80	**11.38**	**9.59**
Qwen3 4B	Original	38.33	60.09	23.13	29.10	22.56	3.02	23.13	29.10	31.00	26.12
	SFT	46.67	61.72	30.71	30.50	24.74	2.16	28.34	31.80	<u>33.82</u>	30.34
	DFT	35.00	62.91	34.00	32.40	26.45	5.60	30.16	33.40	32.49	<u>32.49</u>
	iw-SFT	40.00	62.31	28.79	31.00	24.83	2.59	29.34	32.60	32.43	30.43
	Random	38.33	60.98	31.35	30.50	25.69	0.86	29.34	31.30	31.46	30.62
	VCORE	48.33	66.17	34.64	34.30	25.97	3.88	30.25	32.30	**36.09**	**32.87**
Qwen3 8B	Original	43.33	60.09	24.86	34.30	25.21	4.31	24.86	34.30	33.24	29.58
	SFT	48.33	63.80	35.47	34.00	26.07	4.74	34.92	37.90	<u>35.74</u>	35.07
	DFT	41.67	65.13	38.67	37.90	29.38	6.03	35.10	37.50	35.55	**37.29**
	iw-SFT	45.00	61.72	35.37	34.60	27.20	5.60	35.47	35.90	34.88	<u>35.34</u>
	Random	43.33	61.57	35.47	36.20	25.59	4.31	33.46	35.90	33.70	35.26
	VCORE	45.00	64.84	35.01	36.80	28.63	4.74	33.82	35.40	**35.80**	<u>35.26</u>
Qwen3 32B	Original	43.33	64.99	41.13	43.90	27.30	5.60	41.13	43.90	35.30	42.52
	SFT	38.33	63.80	46.16	39.80	31.94	9.91	48.54	40.60	35.99	43.78
	DFT	43.33	67.66	54.48	46.50	37.82	10.34	49.91	47.40	<u>39.79</u>	**49.57**
	iw-SFT	40.00	63.65	43.33	39.40	31.94	6.90	45.43	42.40	35.62	42.64
	Random	40.00	62.31	45.80	40.30	35.55	9.91	47.71	45.50	36.94	44.83
	VCORE	51.67	68.55	49.91	45.00	35.45	9.48	45.70	43.10	**41.29**	<u>45.93</u>

Table 1: **Main Results.** Accuracy (%) of different methods on **in-domain** (AIME, Olympiad, LCB, OJBench) and **out-of-domain** (RBench, SGPQA-1k) benchmarks across Math and Code. Best and second-best except **Original** are shown in **bold** and <u>underline</u>, respectively.

2025 (**AIME**) (Art of Problem Solving Community, 2025) and math subset of OlympiadBench (**Olympiad**) (He et al., 2024) for math, OJBench (**OJBench**) (Wang et al., 2025b) and LiveCodeBench(v6) (**LCB**) (Jain et al., 2025) for coding. For **out-of-domain** evaluation, we use R-Bench-T (**RBench**) (Guo et al., 2025b) and a 1k-sample subset of SuperGPQA (**SGPQA-1k**) (Team et al., 2025b). We run inference using vLLM v1 (Kwon et al., 2023) with a maximum generation length of 8192 tokens. For all the benchmarks, we use greedy decoding and report Pass@1 as the metric. See Appendix B.3 for more detailed settings.

Baselines. We compare against two recent reweighting-based methods: **DFT** (Wu et al., 2025) and **iw-SFT** (Qin and Springenberg, 2025) We also include the **Original** model (without any fine-tuning) and standard **SFT**, which uses the full CoT supervision signal. In the **Random** baseline, 80% of CoT tokens are randomly dropped during training to simulate partial supervision.

Implementation Details. All the training experiments are conducted on four RTX PRO 6000 Blackwell GPUs using the LLaMA-Factory framework (Zheng et al., 2024), with a batch size of 32. We apply one epoch of LoRA fine-tuning using the AdamW optimizer with a cosine learning rate schedule. The learning rate is set to $2e{-}5$ for Qwen3 models and $2e{-}4$ for LLaMA3.1-8B-Instruct. The LoRA rank is set to 8 for Qwen3 and 64 for LLaMA, with the LoRA alpha fixed at twice the rank in both cases. In the algorithmic setting of VCORE, for each parameter update we randomly select a batch B' from the training set. The hyperparameters ϵ and τ are tuned specifically for each model. Complete training configurations are provided in Appendix B.2.

5.2 Main Results (RQ1)

Obs 1: VCORE demonstrates the best overall performance, achieving strong in-domain accuracy and robust out-of-domain generalization across different models and domains. As shown in Table 1, averaged across all evaluation results, VCORE obtains the best overall score (31.03), surpassing alternative approaches such as SFT (28.97), DFT (29.99), iw-SFT (28.35), and Random (28.78).

(a) Supervised Set Size

Size	Olympiad	SGPQA-1k
Original	60.09	29.10
4k	65.13 ∣ 62.17	32.70 ∣ 32.70
8k	63.35 ∣ 62.17	32.80 ∣ 29.90
16k	62.46 ∣ 61.28	30.80 ∣ 30.30
32k	62.02 ∣ 62.02	30.60 ∣ 28.80

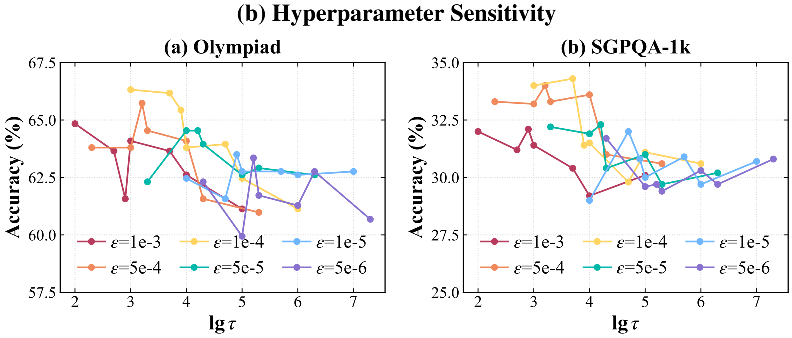

Figure 2: **Component Analysis and Ablation.** (a) Impact of supervised set size on in-domain (Olympiad) and out-of-domain (SGPQA-1k) accuracy for **VCORE | DFT**; (b) Hyperparameters: reweighting temperature τ and probing scale ϵ. All results use Qwen3-4B. Metrics are accuracy (%) on Olympiad (in-domain) and SGPQA-1k (out-of-domain).

VCORE maintains robust cross-domain reasoning without sacrificing supervised task performance. This balance demonstrates that variance-controlled reweighting successfully avoids the degradation often observed in long CoT supervision.

Obs 2: VCORE yields larger improvements on smaller and less capable models, with gains scaling positively with the strength of larger models. Relative to DFT, VCORE achieves consistently better results on smaller-scale and lower-capacity models. As shown in Table 1, for LLaMA-Instruct, the scores increase from 5.54/7.17 (DFT) to 11.38/9.59 (VCORE), while for Qwen-4B, the results rise from 32.49/32.49 to 36.09/32.87. This indicates that VCORE's optimization-oriented formulation adapts more effectively to limited model capacity than DFT's entropy-minimization perspective. At the same time, as model size increases from Qwen-8B to Qwen-32B, the average performance improvement over the original baseline grows from +4.12 to +4.70, indicating stronger scaling benefits. This suggests that VCORE not only rescues underperforming models but also synergizes with stronger architectures to further enhance reasoning.

5.3 Component Analysis and Ablations (RQ2)

Obs 3: As the training dataset scales up, VCORE consistently maintains its advantage over DFT method. As shown in Figure 2 (a), when the supervised set expands from 4k to 32k, VCORE achieves higher accuracy on both Olympiad and SGPQA-1k benchmarks. This indicates that VCORE generalizes robustly with larger training data sizes.

Obs 4: Optimization-derived reweighting hyperparameter sensitivity. We evaluate how two key components of VCORE's reweighting mechanism affect performance: the reweighting temperature τ, which controls the shape of the supervision distribution $q^*(t)$, and the probing scale ϵ, which determines the accuracy of token utility estimation. Figure 2 (b) summarizes results on Olympiad (in-domain), and SGPQA-1k (out-of-domain), using Qwen3-4B. We can find that VCORE remains relatively stable across a broad range of τ and ϵ values, showing only mild fluctuations. This suggests that VCORE's optimization-driven weighting scheme is robust to hyperparameter variation and does not heavily rely on fine-tuning specific parameter choices. Performance generally peaks around moderate settings (e.g., ϵ around $1e{-}4$, $\log \tau$ around 4), after which excessive reweighting leads to accuracy degradation. This is because very large τ makes the weights approach a nearly uniform distribution, while very small τ overly concentrates the weights on a few tokens, both of which undermine the effectiveness of VCORE's adaptive optimization process.

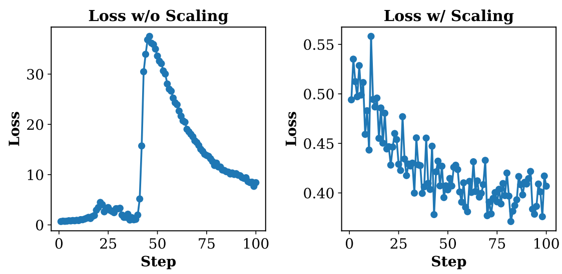

Figure 3: **Loss Scaling.** Loss curves of Qwen3-4B trained on the math domain with and without loss scaling ($\epsilon = 1e{-}4$, $\tau = 5e3$).

Obs 5: Variance control is critical for stabilizing sharp reweighting and ensuring reliable convergence. Figure 3 demonstrates the instability that arises under a sharp temperature setting:

without proper scaling, the loss exhibits a transient spike before stabilizing. This highlights that a critical tension-sharp reweighting improves focus but amplifies gradient variance, which can destabilize training. VCORE mitigates this by introducing an adaptive descent scaling factor that aligns the update variance with that of uniform supervision. As a result, training becomes smoother, more stable, and reproducible. This confirms that variance control is essential for safely leveraging optimization-aligned supervision.

5.4 Discussions: Practical Implications and Limitations (RQ3)

Method	Olympiad		SGPQA-1k	
	Before RL	After RL	Before RL	After RL
DFT	62.91	65.13	32.40	28.30
VCORE	**66.17**	**67.51**	**34.30**	**33.90**

(a) Results on Qwen3-4B.

Method	Olympiad		SGPQA-1k	
	Before RL	After RL	Before RL	After RL
DFT	**65.13**	67.95	**37.90**	38.00
VCORE	64.84	**68.84**	36.80	**38.60**

(b) Results on Qwen3-8B.

Table 2: **RL Results.** Performance before and after RL testing on Olympiad and SGPQA-1k using DFT and VCORE. We highlight the best performance in **bold**.

Obs 6: VCORE offers a more capable foundation model to support reasoning tasks in reinforcement learning. To assess the downstream reasoning capabilities of different post-SFT models, we select the DFT and VCORE variants of Qwen-4B and Qwen-8B and fine-tune them using GRPO (Shao et al., 2024b) for 200 steps on a subset of the BigMath (Albalak et al., 2025) dataset. Evaluation is performed on Olympiad and SGPQA-1k, following the same setup as in Section 5.1. Further details of the RL setup are presented in Appendix C. As shown in Table 2, VCORE exhibits a clear performance improvement over DFT after RL training, despite starting from a slightly lower baseline. The observation indicates that models initialized with VCORE possess a higher upper bound of RL performance. We conjecture that this is because the DFT model tends to reduce the overall generation entropy, thereby limiting the policy's exploration capability during RL. This restriction limits the discovery of improved reasoning trajectories, revealing yet another advantage of VCORE over DFT in fostering reasoning generalization.

Method	GSM8K	MATH500
Original	95.15	90.4
SFT	93.63 (-1.52)	92.2 (+1.8)
DFT	94.69 (-0.46)	92.6 (+2.2)
iw-SFT	93.18 (-1.97)	90.8 (+0.4)
Random	93.56 (-1.59)	92.4 (+2.0)
VCORE	94.16 (-0.99)	92.8 (+2.4)

Table 3: Performance comparison of Qwen3-8B with different SFT strategies on GSM8K and MATH500. Numbers in parentheses indicate the performance change relative to *Original*.

Obs 7: Long CoT SFT methods such as VCORE may exhibit performance degradation when the task is relatively simple. To assess reasoning ability under different difficulty levels, we employ two mathematical reasoning benchmarks: GSM8K (Cobbe et al., 2021) and MATH500 (Lightman et al., 2023). GSM8K is composed of grade-school level arithmetic word problems that typically require only a few steps of numerical reasoning. In contrast, MATH500 contains a subset of competition-level problems from the MATH dataset, demanding deeper symbolic manipulation, abstract reasoning, and multi-step derivations. Using the finetuned Qwen-8B model and same evaluation settings in Section 5.1, we observe from Table 3 that long CoT SFT slightly degrades performance on easy problems but yields consistent gains on harder benchmarks that demand multi-step reasoning. This indicates that our long CoT SFT approach primarily benefits reasoning-intensive tasks rather than simple question answering.

6 Conclusion

In this paper, we present an in-depth investigation into improving the reasoning capabilities of LLMs through long CoT supervised fine-tuning. We introduce VCORE, a variance-controlled, optimization-based reweighting framework. Going beyond heuristic token-weighting methods, we formulate VCORE as an optimization problem that identifies the optimal token importance distribution by maximizing expected loss descent under SGD. Experiments on mathematical and coding benchmarks demonstrate that VCORE significantly enhances reasoning performance, particularly on complex tasks. These results bridge the gap between heuristic SFT practices and optimization-theoretic principles, offering a principled path toward building more generalizable reasoning models.

Limitations

Our study is subject to computational and time constraints, which restricted the training corpus to the OpenMathReasoning and OpenCodeReasoning datasets, both derived from long CoT annotations generated by DeepSeek. We have not yet explored more diverse datasets or long CoT data produced by other state-of-the-art reasoning models, which could potentially reveal different generalization behaviors. We consider this an important avenue for future research.

There is also a potential failure mode where reweighting overemphasizes spurious patterns that leak information about the final answer. While rare, this mode can amplify dataset artifacts or annotation bias. Addressing this may require integrating additional regularization (e.g., dropout masking, answer prefix control) into the utility computation.

References

AI@Meta. 2024. Llama 3 model card.

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. 2025. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *Preprint*, arXiv:2502.17387.

Art of Problem Solving Community. 2025. Aime problems and solutions. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025a. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, and 1 others. 2025b. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*.

Daewon Choi, Jimin Lee, Jihoon Tack, Woomin Song, Saket Dingliwal, Sai Muralidhar Jayanthi, Bhavana Ganesh, Jinwoo Shin, Aram Galstyan, and Sravan Babu Bodapati. 2025. Think clearly: Improving reasoning via redundant token pruning. *arXiv preprint arXiv:2507.08806*.

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

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

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

Meng-Hao Guo, Jiajun Xu, Yi Zhang, Jiaxi Song, Haoyang Peng, Yi-Xuan Deng, Xinzhi Dong, Kiyohiro Nakayama, Zhengyang Geng, Chen Wang, and 1 others. 2025b. R-bench: Graduate-level multidisciplinary benchmarks for llm & mllm complex reasoning evaluation. *arXiv preprint arXiv:2505.02018*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *Preprint*, arXiv:2402.14008.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. 2018. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Zhenyu Hou, Xin Lv, Rui Lu, Jiajie Zhang, Yujiang Li, Zijun Yao, Juanzi Li, Jie Tang, and Yuxiao Dong. 2025. Advancing language model reasoning through reinforcement learning and inference scaling. *arXiv preprint arXiv:2501.11651*.

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

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thir-*

teenth International Conference on Learning Representations.

Kaixuan Ji, Jiafan He, and Quanquan Gu. 2024. Reinforcement learning from human feedback with active queries. *arXiv preprint arXiv:2402.09401*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Bespoke Labs. 2025. Bespoke-stratos: The unreasonable effectiveness of reasoning distillation. www.bespokelabs.ai/blog/bespoke-stratos-the-unreasonable-effectiveness-of-reasoning-distillation. Accessed: 2025-01-22.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. 2025. Compressing chain-of-thought in llms via step entropy. *arXiv preprint arXiv:2508.03346*.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. 2023. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv preprint arXiv:2310.10505*.

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

Zicheng Lin, Tian Liang, Jiahao Xu, Qiuzhi Lin, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. 2024. Critical tokens matter: Token-level contrastive estimation enhances llm's reasoning capability. *arXiv preprint arXiv:2411.19943*.

Elita Lobo, Chirag Agarwal, and Himabindu Lakkaraju. 2025. On the impact of fine-tuning on chain-of-thought reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11679–11698, Albuquerque, New Mexico. Association for Computational Linguistics.

Renjie Luo, Jiaxi Li, Chen Huang, and Wei Lu. 2025. Through the valley: Path to effective long cot training for small language models. *Preprint*, arXiv:2506.07712.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. 2025. Aimo-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. *arXiv preprint arXiv:2504.16891*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

NVIDIA. 2025. Opencodereasoning-2. https://huggingface.co/datasets/nvidia/OpenCodeReasoning-2. Accessed: 2025-08-03.

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

Chongli Qin and Jost Tobias Springenberg. 2025. Supervised fine tuning on curated data is reinforcement learning (and can be improved). *arXiv preprint arXiv:2507.12856*.

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

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024a. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024b. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

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

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, and 1 others. 2025a. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

M-A-P Team, Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Kang Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, Kaixing Deng, Shuyue Guo, Shian Jia, Sichao Jiang, Yiyan Liao, Rui Li, Qinrui Li, and 76 others. 2025b. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *Preprint*, arXiv:2502.14739.

NovaSky Team. 2025. Sky-t1: Train your own o1 preview model within $450. https://novasky-ai.github.io/posts/sky-t1. Accessed: 2025-01-09.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965.

Jean Vassoyan, Nathanaël Beau, and Roman Plaud. 2025. Ignore the kl penalty! boosting exploration on critical tokens to enhance rl fine-tuning. *arXiv preprint arXiv:2502.06533*.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2023. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*.

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Zhexu Wang, Yiping Liu, Yejie Wang, Wenyang He, Bofei Gao, Muxi Diao, Yanxu Chen, Kelin Fu, Flood Sung, Zhilin Yang, Tianyu Liu, and Weiran Xu. 2025b. Ojbench: A competition level code benchmark for large language models. *Preprint*, arXiv:2506.16395.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Sean Welleck, Amanda Bertsch, Matthew Finlayson, Hailey Schoelkopf, Alex Xie, Graham Neubig, Ilia Kulikov, and Zaid Harchaoui. 2024. From decoding to meta-generation: Inference-time algorithms for large language models. *arXiv preprint arXiv:2406.16838*.

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

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

Yongliang Wu, Yizhou Zhou, Zhou Ziheng, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. 2025. On the generalization of sft: A reinforcement learning perspective with reward rectification. *arXiv preprint arXiv:2508.05629*.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*.

Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, and 1 others. 2025. Redstar: Does scaling long-cot data unlock better slow-reasoning systems? *arXiv preprint arXiv:2501.11284*.

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, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*.

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

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, and 1 others. 2025a. A survey on test-time scaling in large language models: What, how, where, and how well? *arXiv preprint arXiv:2503.24235*.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025b. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, and 1 others. 2023. Secrets of rlhf in large language models part i: Ppo. *arXiv preprint arXiv:2307.04964*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and 1 others. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Zhanke Zhou, Rong Tao, Jianing Zhu, Yiwen Luo, Zengmao Wang, and Bo Han. 2024. Can language models perform robust reasoning in chain-of-thought prompting with noisy rationales? *Advances in Neural Information Processing Systems*, 37:123846–123910.

A Related Work

The rise of test-time scaling and long CoT. Recent advances in LLMs have highlighted the importance of *test-time scaling* (Snell et al., 2024; Welleck et al., 2024; Zhang et al., 2025a), i.e., improving reasoning performance by allocating more inference-time compute rather than solely scaling model size or training resources. A practical instance of this idea is *long chain-of-thought (long CoT)*: LLMs allocate more compute to generate longer reasoning chains at inference. Recent LLMs—OpenAI-o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025a), Kimi k1.5 (Team et al., 2025a)—pursue this direction by scaling CoT lengths and show strong reasoning performance on tasks such as challenging math problem solving and code generation benchmarks. Compared with short CoT, long CoT enables deeper reasoning, more extensive exploration, and feasible reflection (Chen et al., 2025a), thereby supporting more complex reasoning tasks.

Enhancing LLM reasoning ability. While CoT prompting (Wei et al., 2022) can improve reasoning by explicitly encouraging models to articulate intermediate steps in natural language, it still struggles on complex reasoning problems. Consequently, recent work has focused on strengthening models' intrinsic reasoning via post-training or test-time methods:

- **RL for reasoning.** Reinforcement learning has emerged as an effective post-training stage for eliciting multi-step reasoning in LLMs (Ouyang et al., 2022; Lightman et al., 2023; Guo et al., 2025a; Wen et al., 2025; Hou et al., 2025). On the algorithmic side, most work adopts policy-gradient methods such as REINFORCE (Williams, 1992), PPO (Schulman et al., 2017) and LLM-tailored variants such as ReMax and (Li et al., 2023), GRPO (Shao et al., 2024b), DAPO (Yu et al., 2025), etc. In parallel, some preference optimization (PO) methods (e.g., DPO (Rafailov et al., 2023), KTO (Ethayarajh et al., 2024), IPO (Azar et al., 2024), CPO (Xu et al., 2024)) optimize supervised objectives built from pairwise comparisons or binary accept/reject signals, avoiding on-policy rollouts. Orthogonal to the optimization algorithm, we categorize methods by reward source: (1) *Outcome-reward* RL, which directly optimizes final answers; a prominent subfamily is RL with verifiable rewards (RLVR) for math/coding, where unit tests or checkers define the reward (Guo et al., 2025a; Lambert et al., 2024); and (2) *Process-reward* RL, which performs step-level credit assignment via process reward models (PRMs) to shape the reasoning trajectory and improve reasoning faithfulness (Lightman et al., 2023; Wang et al., 2023; Zhang et al., 2025b).

- **SFT for reasoning:** However, adopting RL may sometimes be cumbersome—sensitive to hyperparameters (Henderson et al., 2018; Zheng et al., 2023), resource-intensive (Schulman et al., 2017), and costly in terms of training data collection (Ji et al., 2024; Wang et al., 2023). In comparison, a more straightforward way for enhancing reasoning ability is to distill long reasoning traces into LLMs via SFT. Recent works show that training on curated long CoT data—either distilled from stronger teachers or manually constructed—can endow models with robust slow-thinking behaviors (Team, 2025; Muennighoff et al., 2025; Xu et al., 2025; Labs, 2025; Ye et al., 2025). Compared to SFT on short CoT, SFT on long CoT offers some practical benefits: a higher performance ceiling, better generalization, and larger downstream gains when used to initialize RL (Yeo et al., 2025).

- **Test-time methods:** A complementary line of work improves reasoning *during inference*. Approaches roughly fall into categories, including but not limited to: (i) prompting-based strategies that encourage stepwise thinking or decomposition—zero-/few-shot chain-of-thought (Wei et al., 2022), least-to-most prompting (Zhou et al., 2022), and ReAct-style reasoning–acting (Yao et al., 2023b); (ii) sampling-and-aggregation schemes that draw multiple rationales and then vote or rerank, e.g., self-consistency (Wang et al., 2022); (iii) search/planning over intermediate states, such as tree-structured exploration (Yao et al., 2023a); and (iv) self-reflection and debate that iteratively critique and revise candidate chains (Madaan et al., 2023). In practice, such test-time strategies are often combined and are complementary to SFT and RL for eliciting multi-step reasoning.

We remark that our method falls into the *SFT for reasoning* category and our work focuses on modifying the SFT phase itself to endow models with stronger generalization ability on reasoning tasks.

Token-Level Reweighting in SFT To retain the simplicity of SFT yet benefit from RL-induced reasoning improvements, recent studies modify the SFT objective to narrow its gap with RL. Specifically, Dynamic Fine-Tuning (DFT) (Wu et al., 2025) and importance-weighted SFT (iw-SFT) (Qin and Springenberg, 2025) pursue this via token-level loss reweighting. DFT reinterprets standard SFT as a biased policy update that over-concentrates on low-probability tokens; accordingly, it neutralizes that bias by rescaling the token-level loss. iw-SFT shows that SFT on curated/-filtered data optimizes a lower bound to an RL objective; accordingly, it tightens that bound with explicit importance weights relative to a reference or current policy. Since our method can be regarded as a hard/sparse version of token-level loss reweighting, we include DFT and iw-SFT as our baselines.

B Experimental Details of Main Results

B.1 Dataset Curation for supervised tasks

For CoT supervised fine-tuning, we use two datasets: OpenMathReasoning[1] and OpenCodeReasoning[2], which correspond to the mathematics, and coding domains, respectively. We summarize the details of sampling and processing procedure as follows:

OpenMathReasoning. From the 3.2M samples in the `cot` split, we extract a subset that satisfies the following conditions: `problem_type = "has_answer_extracted"` and `generation_model = "DeepSeek-R1"`. We then use **math_verify**[3] to rigorously check whether the answers in the generated CoT content are equivalent to the expected answers. Based on this verified subset, we randomly sample 3,200 examples for training the Qwen3 series and 32,000 examples for training LLaMA3.1-8B-Instruct. To adapt the data for CoT training, we augment the original questions with new prompts designed for CoT reasoning. The specific format is as follows:

[1] https://huggingface.co/datasets/nvidia/OpenMathReasoning
[2] https://huggingface.co/datasets/nvidia/OpenCodeReasoning-2
[3] https://github.com/huggingface/Math-Verify

Math CoT Training Data Template

You are a helpful and accurate assistant for solving the following math problem:
{ORIGINAL QUESTION}

Please reason step by step, and put the correct answer within \boxed{} at last.

OpenCodeReasoning. From the 942K samples in the cpp split, we extract the subset that satisfies the condition `judgement = "right"`. Based on this subset, we randomly sample 3,200 examples for training the Qwen3 series and 32,000 examples for training LLaMA3.1-8B-Instruct. For the training data templates, we take inspiration from the prompt templates in `open-r1/codeforces`[4] and design our own training templates as follows:

Code CoT Training Data Template

You are an expert competitive programmer. You will be given a problem statement, test case constraints, and example test inputs and outputs.
Please reason step by step about the solution (that must respect memory and time limits), then provide a complete implementation in c++17.

Your solution must read input from standard input (cin) and write output to standard output (cout).
Do not include any debug prints or additional output.

Put your final solution within a single code block:
```cpp
&lt;your code here&gt;
```

Problem
{ORIGINAL QUESTION}

Now solve the problem and return the code.

[4]https://huggingface.co/datasets/open-r1/codeforces

Hyperparameter	Qwen3	LLaMA3.1-8B-Instruct
batch size	32	32
learning rate	2e-5	2e-4
training steps	100	1000
LoRA target	all	all
LoRA rank / alpha	8 / 16	64 / 128
LoRA dropout	0.10	0.05
lr schedule	cosine	cosine
warmup ratio	0.10	0.05
optimizer	AdamW	AdamW
seed	42	42
data type	bf16	bf16
cutoff length	16384	16384

Table 4: Hyperparameter settings for Qwen3 and LLaMA3.1-8B-Instruct.

B.2 Training Details

All baselines and our method are implemented on top of the `LLaMA-Factory`[5] framework. Most hyperparameters are shared across these methods, as summarized in Table 4.

Hyperparameters of VCORE As shown in Figure 2, we conduct a small-scale grid search over $\epsilon \in \{1e{-}4, 1e{-}5\}$ and $\tau \cdot \epsilon \in \{0.5, 0.8\}$, resulting in four configurations. We train each model under these settings and report the best-performing one in Table 5.

Model	Math		Code	
	ϵ	$\tau \cdot \epsilon$	ϵ	$\tau \cdot \epsilon$
LLaMA3.1-8B Instruct	1e-5	0.8	1e-5	0.8
Qwen3-4B	1e-4	0.5	1e-4	0.8
Qwen3-8B	1e-5	0.5	1e-5	0.5
Qwen3-32B	1e-4	0.8	1e-4	0.8

Table 5: Selected hyperparameters $(\epsilon, \tau \cdot \epsilon)$ for each model and domain.

The implementation details of Method Random. In the **Random** baseline method, we retain only 20% of the original supervision tokens in total. The supervision on the final answer tokens (those inside `\boxed{...}`) is always preserved. Once these tokens are fixed, we randomly sample from the remaining supervision tokens such that the overall proportion of preserved tokens (answer plus non–answer) amounts to 20%. All other tokens are excluded from the loss.

The purpose of designing this algorithm is to investigate the effect of sparse supervision on SFT. The method essentially performs a discrete binary weighting of supervision tokens, thereby allowing us to ablate and examine how reducing the amount

[5]https://github.com/hiyouga/LLaMA-Factory

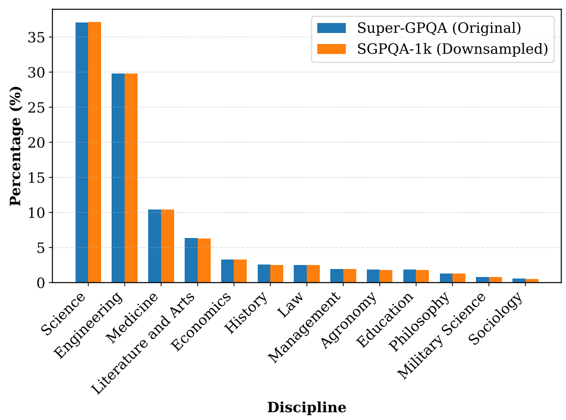

Figure 4: Construction of GPQA-1k.

of supervision signal influences the training dynamics and overall performance.

B.3 Evaluation Details

We select two groups of benchmarks, each consisting of two domain-specific and two comprehensive tasks, resulting in a total of six benchmarks to thoroughly evaluate the generalization ability of different methods. Detailed information for each benchmark is provided in Table 6.

Specifically, for computational efficiency, we adopt a 1,000-sample i.i.d. subset from the SuperGPQA, referred to as SGPQA-1k. We ensure that its distribution of different disciplines remains consistent with the original dataset. Figure 4 illustrates the distribution before and after downsampling.

For OJBench, evaluation is performed using its original problem format. For other non-multiple-choice benchmarks, we adopt the same instruction templates as used in the corresponding training domain during inference. For multiple-choice benchmarks, including the two comprehensive benchmarks, we use the template provided below. All evaluations are conducted on 8×NVIDIA RTX PRO 6000 Blackwell GPUs.

We evaluate the models using the vLLM[6] framework. The detailed hyperparameter settings for inference are provided in Table 7.

We extract the predicted answers from model outputs using regular expression matching on the `\boxed{}` format. For AIME and Olympiad, we further apply **math_verify** to ensure answer equivalence. For OJBench[7], LiveCodeBench[8] , we adopt the official repositories for evaluation. For two comprehensive benchmarks, we directly verify answers using exact match.

Multiple-Choice Benchmark Evaluation Template

You are a helpful and accurate assistant for solving the following multiple-choice question:
{ORIGINAL QUESTION}
Options are:
(A): {OPTION A}
(B): {OPTION B}
...

Please reason step by step, and put the correct answer within \boxed{ } at last.

C Experimental Details of Reinforcement Learning

We conduct the reinforcement learning experiments using the VERL[9] framework with the GRPO(Shao et al., 2024a) algorithm. For training, we randomly sample 16,800 examples from the Big-Math[10] dataset, which is the largest open-source dataset of high-quality mathematical problems, curated specifically for RL training in LLMs. We perform additional full-parameter RL training on the Qwen3-4B and Qwen3-8B models obtained from the main experiments. To ensure comparability, we use identical hyperparameter configurations across model sizes and initialization strategies (DFT and VCORE). The detailed hyperparameter settings are provided in Table 8.

D Experimental Details of GSM8K and MATH500 Evaluation

We evaluate our models on the test split of GSM8K[11] and MATH500[12] . The evaluation is conducted using vLLM, and all other testing configurations remain identical to those described in Table 7.

[6]https://github.com/vllm-project/vllm/releases/tag/v0.9.2
[7]https://github.com/He-Ren/OJBench
[8]https://github.com/LiveCodeBench/LiveCodeBench
[9]https://github.com/volcengine/verl
[10]https://huggingface.co/datasets/SynthLabsAI/Big-Math-RL-Verified
[11]https://huggingface.co/datasets/openai/gsm8k
[12]https://github.com/openai/prm800k

Benchmark	Size	Year	Task Type	Metric	Subset	Source
AIME	60	2024 2025	Open-ended QA	Acc@1	Full	AoPS 2024 I AoPS 2024 II AoPS 2025 I AoPS 2025 II
Olympiad	674	2024	Open-ended QA	Acc@1	OE_TO_maths_en_COMP	HF (OlympiadBench)
LCB (v6)	1055	2024	Code generation	Pass@1	release_v6	HF (LiveCodeBench)
OJBench	232	2025	Code generation	Pass@1	Full	HF (OJBench)
RBench	1094	2025	Multiple-choice QA	Acc@1	rbench-t_en	HF (R-Bench)
SGPQA-1k	1000	2025	Multiple-choice QA	Acc@1	1k samples	HF (SuperGPQA)

Table 6: Benchmarks used for evaluation.

Hyperparameter	Value
temperature	0
top_p	1.0
top_k	-1
batch_size	512
VLLM_USE_V1	True
seed	42
enable_thinking	True
max_new_tokens	8192

Table 7: Inference hyperparameters used in evaluation.

Hyperparameter	Value
algorithm	GRPO
train_batch_size	128
max_prompt_length	4096
max_response_length	8192
total_epochs	2
n_gpus_per_node	8
learning rate	5e-6
lr_warmup_steps_ratio	0.1
warmup_style	cosine
ppo_mini_batch_size	32
ppo_micro_batch_size_per_gpu	2
entropy_coeff	0.001
use_kl_loss	True
kl_loss_coef	0.02
kl_loss_type	low_var_kl
use_kl_in_reward	False
rollout.n	4
rollout.max_model_len	8192

Table 8: Hyperparameters used in RL training.