SALT: When More Rollouts Don't Help in Group-Based Policy Optimization and How to Make Them Matter

Powei Chang[*,◇]
Bilibili Inc.
Fudan University

Jinpeng Zhang[*,⚐]
Bilibili Inc.

Chaoqun Sun[◇]
Bilibili Inc.
Fudan University

MiniWell Tsao[◇]
Bilibili Inc.
Fudan University

Lianrui Li[◇]
Bilibili Inc.
Zhejiang University

Jianxiang Xiang
Bilibili Inc.

Chenyu Wang
Bilibili Inc.

Yukang Gao
Bilibili Inc.

Dongying Kong[✉]
Bilibili Inc.

Abstract

Reinforcement learning with verifiable rewards (RLVR) often adopts GRPO-style group-relative updates, sampling multiple rollouts per prompt to construct normalized learning signals. However, merely increasing the number of rollouts does not reliably strengthen learning: under GRPO-style group normalization, per-rollout policy-gradient features can concentrate into a low-rank, signed geometry, causing substantial cancellation during aggregation and weakening the effective update. We address this failure mode with ***SALT***, a Subspace-Adaptive geometry pLug-in componenT that uses sample-wise gradient geometry to reweight the coefficients of group-relative updates. ***SALT*** estimates a dominant spectral subspace from the mini-batch Gram geometry, decomposes group-relative coefficients into dominant and spectral-tail channels, and adaptively reweights the tail channel when signed cancellation is severe. Across diverse reasoning-oriented RLVR benchmarks and model scales, ***SALT*** improves effective update geometry and performance without modifying the reward model or the rollout sampling procedure (Code).

1 Introduction

Reinforcement learning with verifiable rewards (RLVR) has become a central recipe for improving the reasoning ability of large language models (LLMs), as it optimizes model-generated solutions using automatically checkable outcome signals rather than subjective preference labels [42, 31].

Building on PPO-style policy optimization [40], Group Relative Policy Optimization (GRPO) and its variants improve scalability by replacing the critic with group-relative normalization over multiple rollouts, and are now widely used in reasoning-oriented RL pipelines [10, 54]. However, simply increasing the number of rollouts often yields sharply diminishing returns, suggesting a structural inefficiency in how group-relative updates aggregate learning signals [19, 49].

[*]Equal contribution; [⚐] project leader; [✉] corresponding author.
[◇]Work done during an internship at Bilibili Inc.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Prior work attributes this bottleneck to insufficient exploration and limited response diversity within rollout groups [16]. Accordingly, existing remedies rely on policy-side signals, such as entropy-based, KL-based, or confidence-based regularization, to mitigate the loss of response diversity and stabilize group-based training [57, 9, 47]. This leads to our pivotal research question:

> Beyond maintaining response diversity, when do additional rollouts *actually translate* into stronger and more informative *group-relative updates*?

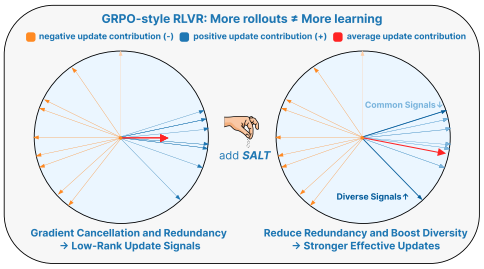

Figure 1: Illustration of rollout inefficiency in GRPO-style RLVR. Per-rollout gradients exhibit signed, low-rank redundancy, so group-normalized aggregation cancels much of the learning signal. ***SALT*** reweights dominant and spectral-tail channels to reduce signed cancellation and strengthen effective updates.

In this work, we take an **update-space** view of rollout effectiveness. Even when rollouts remain distinct at the sequence level, per-sample policy-gradient features within each prompt group can be highly redundant [4]. As illustrated in Figure 1, these features concentrate in a low-rank, signed geometry and, after group-relative advantage weighting, form opposing update directions that cancel during aggregation. We characterize this geometry by low effective dimensionality (PR) for spectral concentration and low effective sample size (n_{eff}) [22] for weak net update strength after signed aggregation. Together, these diagnostics explain why additional rollouts add compute without proportional learning gains.

This saturation arises from the structure of group-relative optimization. GRPO-style objectives center advantages within each prompt group [42], canceling gradient components shared across rollouts; when per-sample gradient features concentrate into a low-rank, signed geometry, the aggregated update is left to depend on a small *residual* component and can therefore fail to scale with additional rollouts. Thus, rollout inefficiency is not merely a sampling issue, but a failure to aggregate residual, non-redundant signals into effective learning progress.

Motivated by this diagnosis, we propose ***SALT***, a Subspace-Adaptive geometry pLug-in componenT for GRPO-style training that uses mini-batch gradient geometry to reweight group-relative coefficients and reduce signed aggregation cancellation. *SALT* estimates a dominant spectral subspace from mini-batch Gram geometry via a lightweight proxy [17], decomposes group-relative coefficients into dominant and spectral-tail channels, and adaptively reweights the tail channel with a cancellation-aware mixing coefficient. This spectral decomposition is a separately defined geometry-aware surrogate, rather than an identification with the within-prompt mean–residual decomposition above. By sample-wise geometry-based reweighting, *SALT* improves effective update geometry—spectral spread and signed aggregation strength—rather than relying only on policy-side exploration heuristics. We evaluate *SALT* across reasoning RLVR benchmarks, two model scales, and multiple GRPO-style recipes, assessing both task performance and update-geometry diagnostics. Our contributions are:

• (**Phenomenon**) We diagnose signed low-rank redundancy in group-relative RLVR: within- and across-group policy-gradient features concentrate in a low-dimensional geometry.

• (**Method**) We propose ***SALT***, a subspace-adaptive plug-in that estimates a dominant spectral subspace and reweights group-relative coefficients through dominant and spectral-tail channels.

• (**Effectiveness**) We validate ***SALT*** with extensive experiments and ablations, demonstrating consistent gains and improved effective update directions.

2 Gradient Redundancy

2.1 Preliminaries

Notation. We view an auto-regressive language model with parameters θ as a policy π_θ [45]. Given a prompt x and response $y = (y_1, \ldots, y_{|y|})$,

$$\pi_\theta(y \mid x) = \prod_{t=1}^{|y|} \pi_\theta(y_t \mid x, y_{<t}), \tag{1}$$

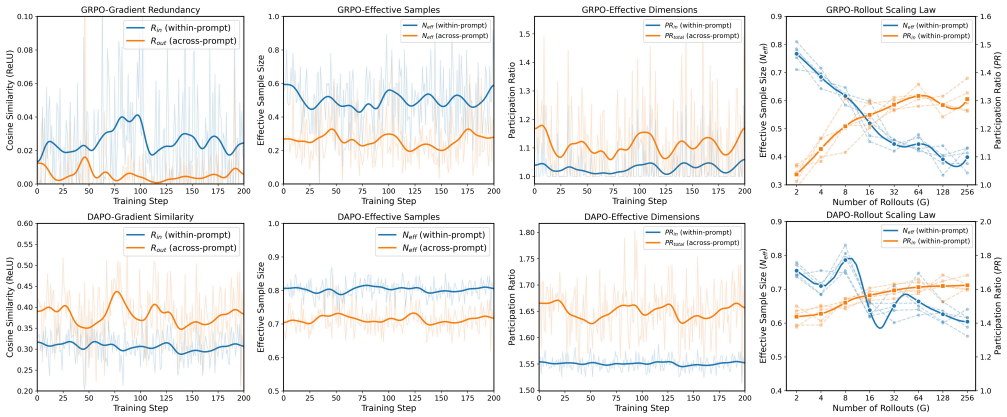

Figure 2: Signed low-rank gradient redundancy in GRPO-style RLVR. (a) **Redundancy dynamics** for GRPO (top) and DAPO (bottom): throughout training, per-sample policy-gradient features concentrate in a low-dimensional subspace, causing strong cancellation and few effective independent update signals despite diverse rollouts. (b) **Rollout scaling**: increasing group size makes PR and n_{eff} saturate rather than scale proportionally, showing that more rollouts need not yield more learning.

with reward $r \in [0, 1]$ assigned to each prompt–response pair (x, y).

Group Relative Policy Optimization (GRPO). We now focus on GRPO [42, 11], which can introduce aggregation biases both within and across groups in our study. In each iteration, GRPO samples B prompts from $P(Q)$; for each prompt q, it then samples a set of G candidate responses $\{o_1, o_2, \ldots, o_G\}$ from the current policy $\pi_{\theta_{\text{old}}}$. Each response o_i is assigned a scalar reward r_i. Rewards within the group are normalized to obtain a response-level advantage [13]:

$$\hat{A}_i = \frac{r_i - \text{mean}(r_1, r_2, \cdots, r_G)}{\text{std}(r_1, r_2, \cdots, r_G)}. \tag{2}$$

It then updates the policy π_θ using a PPO-style clipped objective [40]. Formally, the GRPO objective function is defined as:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\,\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(O|q)} \left[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|}\sum_{t=1}^{|o_i|} \left(\min\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})} \hat{A}_{i,t}, \right.\right.\right. \\ \left.\left.\left. \text{clip}\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}, 1-\epsilon,\ 1+\epsilon \right) \hat{A}_{i,t} \right) - \beta\, \mathbb{D}_{\text{KL}}[\pi_\theta \,\|\, \pi_{\text{ref}}] \right) \right]. \tag{3}$$

More broadly, the rollout-based group optimization paradigm has led to a series of GRPO-style RLVR methods (e.g., DAPO [54]; more formulations in Appendix I). Nevertheless, they are prone to a shared phenomenon, **within-group and cross-group gradient redundancy**, which we study and characterize in the following sections.

2.2 Gradient Redundancy

We motivate our method from a persistent empirical pattern in GRPO-style RLVR: per-sample gradient features concentrate in a low-dimensional geometry, producing far fewer independent update signals than the raw sample count suggests. We quantify this with two complementary diagnostics: a regularized participation ratio (PR) from the batch Gram spectrum for spectral spread, and an effective sample size proxy (n_{eff}) from signed per-sample update contributions for net aggregation strength [53, 22]. We report both within-group and batch-level variants to separate prompt-local redundancy from redundancy persisting across prompts.

2.2.1 Diagnostics

Effective dimensionality. Let $\{g_i\}_{i=1}^m \subset \mathbb{R}^d$ denote gradient features for a collection of m samples (responses). We form the normalized Gram matrix

$$K \triangleq \frac{1}{m} G^\top G \in \mathbb{R}^{m \times m},\ G \triangleq [g_1, \ldots, g_m] \in \mathbb{R}^{d \times m}. \tag{4}$$

Since K is positive semidefinite, it admits eigenvalues $\lambda_1, \ldots, \lambda_m \geq 0$. We quantify the effective dimensionality of the set $g_i i = 1^m$ via the **participation ratio** (PR), defined as the ratio between the squared first spectral moment and the second spectral moment

$$\mathrm{PR}(K) \triangleq \frac{\left(\sum_{i=1}^m \lambda_i\right)^2}{\sum_{i=1}^m \lambda_i^2}. \tag{5}$$

Larger $\mathrm{PR}(K)$ indicates a less concentrated spectrum and thus a higher effective dimensionality.

Effective sample size. Low PR indicates that gradient features concentrate in a low-dimensional subspace, i.e., stronger redundancy, but does not distinguish coherent agreement from cancellation. To quantify the *net strength* of the aggregated update, we use an **effective sample size** proxy [22, 51].

$$n_{\text{eff}}(\{u_i\}_{i=1}^m) \triangleq \frac{\left\| \sum_{i=1}^m u_i \right\|_2^2}{\sum_{i=1}^m \|u_i\|_2^2}, \tag{6}$$

where $u_i = \hat{A}_i \nabla_\theta \log \pi_\theta$ denotes the signed per-sample update.

Alignment. To make the "opposing orientation" structure explicit, we also track a nonnegative alignment statistic **positive-part cosine alignment** [59] computed between per-sample directions

$$r(\{g_i\}_{i=1}^m) \triangleq \frac{2}{m(m-1)} \sum_{1 \leq i < j \leq m} [\cos(g_i, g_j)]_+. \tag{7}$$

where $[x]_+ = \max(x, 0)$. Higher values indicate stronger aligned modes, which can still cancel after signed aggregation when paired with opposing advantage signs.

2.2.2 Empirical observations

Signed low-rank gradient redundancy. Group-based RL aggregates rollouts within each prompt group before averaging across prompts. We therefore report both within- and cross-group diagnostics for GRPO and DAPO under the setup in Section 4.1, with implementation details in Appendix D.1. As shown in Figure 2(a), per-sample gradient features in GRPO-style training exhibit signed low-rank structure: they concentrate in a shared subspace, while signed update directions often oppose the group or batch mean. This reduces spectral spread and causes strong aggregation cancellation, yielding low PR and n_{eff} even with many rollouts. Appendix F.1 shows the same pattern on execution-verifiable code RLVR and additional model families.

Rollout scaling. Prior work shows that increasing rollout count gives diminishing returns [41, 49]. Figure 2 (b) revisits this effect in gradient space: as G increases over $\{2, 4, 8, 16, 32, 64, 128, 256\}$, neither the spectral-spread diagnostic nor n_{eff} grows proportionally. Under group-relative objectives, additional rollouts often add directionally redundant gradients, contributing little independent signal after group-centered aggregation. These observations motivate the next subsection, where a simple decomposition explains why group-centered updates are vulnerable to redundancy.

2.3 Why redundancy hurts GRPO-style training

Although significant redundancy has been demonstrated, why does it still degrade RL training? For clarity, we analyze only the core policy-gradient term in Eq. (3), omitting KL and clipping terms.

Group centering cancels shared gradients. For a prompt q, GRPO samples G rollouts $\{o_{q,i}\}_{i=1}^{G}$. Let $\hat{A}_{q,i}$ be the group-normalized advantage and $s_{q,i} \triangleq \nabla_\theta \log \pi_\theta(o_{q,i} \mid q)$ be the score gradient. The group-averaged policy-gradient contribution is

$$g_q(\theta) \triangleq \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i} s_{q,i}, \qquad \sum_{i=1}^{G} \hat{A}_{q,i} = 0, \tag{8}$$

where the second relation follows from within-group advantage centering [41]. Thus, any perfectly shared component is canceled: if $s_{q,i} \equiv s_q$, then $g_q(\theta) = G^{-1}(\sum_i \hat{A}_{q,i}) s_q = 0$.

More generally, write each score gradient as the sum of a common component and a residual [5]:

$$s_{q,i} = \bar{s}_q + \delta_{q,i}, \quad \bar{s}_q \triangleq \frac{1}{G}\sum_{i=1}^{G} s_{q,i}, \quad \sum_{i=1}^{G} \delta_{q,i} = 0. \tag{9}$$

Substituting this decomposition into the group update gives

$$g_q(\theta) = \frac{1}{G}\Big(\sum_{i=1}^{G} \hat{A}_{q,i}\Big)\bar{s}_q + \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i}\delta_{q,i} = \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i}\delta_{q,i}. \tag{10}$$

Thus, group centering removes the common component and makes the effective update depend on the residual directions, whose magnitude is bounded by

$$\|g_q(\theta)\|_2 \le \frac{1}{G}\left(\sum_{i=1}^{G} \hat{A}_{q,i}^2\right)^{1/2}\left(\sum_{i=1}^{G} \|\delta_{q,i}\|_2^2\right)^{1/2} \lesssim \left(\frac{1}{G}\sum_{i=1}^{G} \|\delta_{q,i}\|_2^2\right)^{1/2}, \tag{11}$$

where the last step uses $\sum_i \hat{A}_{q,i}^2 \approx G$ under unit empirical variance within the group and the derivation is provided in Appendix A.

Therefore, more rollouts help only when they increase the residual gradient energy. Redundant score-gradient directions are removed by group centering and do not strengthen the update. This explains the rollout-scaling failure mode: sequence-level diversity may still leave small residual directions that cancel under signed advantages, reducing n_{eff}. Thus, effective rollout scaling requires non-redundant residual directions. This residual-update bottleneck motivates *SALT*, but the method's mini-batch spectral decomposition introduced next is distinct from the within-prompt mean–residual decomposition analyzed here. We empirically validate the residual-bottleneck prediction in Appendix B.

3 Methodology

Motivated by the redundancy and signed cancellation identified above, *SALT* does not aim to preserve the original GRPO estimator. Instead, it intentionally constructs a geometry-reweighted surrogate. The procedure is summarized in Algorithm 1 in Appendix J.

3.1 Dual-Channel Advantage Reweighting

Let $\mathcal{I} = \{(b,i) : b = 1,\dots,B,\ i = 1,\dots,G\}$ index all responses in the current mini-batch, and denote $m \triangleq |\mathcal{I}| = BG$. Stack the group-normalized advantages into

$$a \triangleq [\hat{A}_{b,i}]_{(b,i)\in\mathcal{I}} \in \mathbb{R}^m. \tag{12}$$

Dominant subspace. Let the normalized batch Gram matrix $K_{\text{total}} \in \mathbb{R}^{m\times m}$ be as defined in Eq. (4). Consider its eigen-decomposition

$$K_{\text{total}} = V\Lambda V^\top, \quad \Lambda = \operatorname{diag}(\lambda_1,\dots,\lambda_m), \tag{13}$$

where $V = [v_1,\dots,v_m]$ is orthonormal. Let $PR_{\text{total}} \triangleq PR(K_{\text{total}})$ be the participation ratio defined in Eq. (5) and (47). We set the dominant-subspace dimension by

$$k_t \triangleq \operatorname{clip}\big(\lfloor PR_{\text{total}} \rceil,\ 1, BG\big), \tag{14}$$

and define $V_k \triangleq [v_1,\dots,v_{k_t}] \in \mathbb{R}^{m\times k_t}$ together with the orthogonal projectors [55]

$$P_k \triangleq V_k V_k^\top, \qquad P_k^\perp \triangleq I - P_k. \tag{15}$$

All quantities above are computed from mini-batch statistics [12, 15].

Two-channel decomposition and reweighting. Motivated by the cancellation diagnosis in Sec. 2.3, we separately define a signed dominant spectral channel together with a one-sided spectral-tail channel:

$$a_{\text{main}} \triangleq P_k a, \qquad a_{\text{exp}} \triangleq P_k^{\perp} [a]_+, \tag{16}$$

where $[a]_+ \triangleq \max(a, 0)$ is applied elementwise. This avoids symmetrically reweighting heterogeneous negative tail coefficients that may reintroduce cancellation, while $P_k a$ preserves group-relative feedback. These dominant/tail spectral channels are not identified with the within-prompt common/residual terms in Sec. 2.3. We validate this method choice in the ablation study in Section 4.3.

Let $\{u_{b,i}\}_{(b,i)\in\mathcal{I}}$ be the signed per-sample update contributions, and let $n_{\text{eff}}(\{u_{b,i}\})$ be the effective sample size proxy already defined in Eq. (6). We set the adaptive exploration mixing coefficient by

$$\alpha_t \triangleq \max\Big(0\ ,\ 1 - n_{\text{eff}}\big(\{u_{b,i}\}_{(b,i)\in\mathcal{I}}\big)\Big). \tag{17}$$

Finally, we form the reweighted advantages

$$a' \triangleq a_{\text{main}} + \alpha_t a_{\text{exp}} = P_k a + \alpha_t P_k^{\perp} [a]_+. \tag{18}$$

We keep the GRPO-style objective unchanged defined in Eq. (3) and replace the original coefficients $\hat{A}_{b,i}$ by the corresponding entries of a'.

3.2 Computational Cost

SALT uses mini-batch gradient geometry, but full-model per-sample gradients are prohibitive for LLMs. We therefore use the LM-head/output-projection gradient as a lightweight proxy, a standard choice for scalable gradient embeddings and influence estimation [2, 34, 32]. The proxy is not an entry-wise approximation to the full gradient; ***SALT* only requires it to preserve the sample-wise geometry used for reweighting.** Table 1 validates this requirement: the proxy preserves *SALT*-relevant geometry against full-gradient and blockwise references, while null baselines fail. Details are in Appendix E.

Table 1: **LM-head proxy preserves *SALT*-relevant geometry.** Results are averaged over validation checkpoints and mini-batches.

Proxy	Gram $\rho\uparrow$	PR err. $\downarrow$	Subspace $\uparrow$
Reference Gradients Baselines			
Full gradient	0.71	0.09	0.78
Blockwise sketch	0.76	0.07	0.81
Last block	0.79	0.06	0.84
Null Baselines			
Permutation null	0.02	0.47	0.18
Noise null	0.00	0.56	0.13

Let the output logits be $z_{b,i,t} = W_{\text{out}} h_{b,i,t} + b_{\text{out}}$, where $h_{b,i,t} \in \mathbb{R}^d$ and $W_{\text{out}} \in \mathbb{R}^{|\mathcal{V}|\times d}$. We use the response-level output-projection gradient as the proxy feature:

$$\nabla_{W_{\text{out}}} \ell_{b,i} = \frac{1}{|o_{b,i}|} \sum_{t=1}^{|o_{b,i}|} \big(\nabla_{z_{b,i,t}} \ell_{b,i,t}\big) h_{b,i,t}^{\top}. \tag{19}$$

Thus, the geometry cost scales with the rollout mini-batch size $m = BG$ rather than the total parameter count: forming the $m \times m$ Gram matrix costs $O(m^2)$ memory, and extracting the top-k_t subspace costs $O(m^2 k_t)$ time with partial eigensolvers. Runtime results are provided in Appendix G.

3.3 *SALT* as a Geometry-Reweighted Surrogate

SALT is not an unbiased reformulation of the original GRPO estimator, but a geometry-reweighted surrogate that trades controlled coefficient distortion for reduced signed cancellation. Let $m = BG$ and $S = [s_1, \ldots, s_m]$. Treating a' as a stop-gradient coefficient, the original update, the *SALT* update, and the corresponding batch surrogate are

$$\hat{g}_{\text{GRPO}} = \frac{1}{m} S a, \quad \hat{g}_{\text{SALT}} = \frac{1}{m} S a', \quad \mathcal{J}_{\text{SALT}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} a_i' \log \pi_\theta(o_i \mid q_i). \tag{20}$$

instead of the original GRPO surrogate with coefficients a_i. Its deviation from GRPO satisfies

$$\hat{g}_{\text{SALT}} - \hat{g}_{\text{GRPO}} = \frac{1}{m} S(a' - a), \qquad \|\hat{g}_{\text{SALT}} - \hat{g}_{\text{GRPO}}\|_2 \le \frac{1}{m} \|S\|_{\text{op}} \|a' - a\|_2. \tag{21}$$

Table 2: Main results on reasoning benchmarks. We compare the GRPO and DAPO algorithms with our method *SALT* across two different training datasets. We report ACC and Pass@8 and results are averaged over 5 runs. **Bold** numbers indicate improvements with *SALT*. Overall, **incorporating *SALT* yields consistent improvements** over GRPO-style algorithms validated by significance tests.

Dataset	Method	AIME24		AIME25		GSM8K		MATH-500		GPQA	
		ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8
Deepseek-Distill-Qwen-1.5B											
∼	Vanilla	29.1	59.3	22.0	42.6	80.3	95.0	85.5	96.4	34.5	82.4
MATH-TRAIN	GRPO	29.3	59.3	25.6	43.3	80.7	94.8	85.6	96.7	35.0	82.2
	+ *SALT*	**32.1**	**62.9**	**27.1**	**47.2**	**83.4**	**96.5**	**87.2**	**98.4**	**38.4**	**85.5**
	DAPO	31.1	59.9	23.9	44.3	81.0	96.1	86.1	97.0	37.0	81.7
	+ *SALT*	30.6	**60.6**	**25.6**	**46.3**	**82.5**	**97.1**	**87.1**	97.0	**39.6**	**83.5**
DAPO-MATH	GRPO	29.1	59.0	23.6	44.2	82.2	95.3	85.9	96.8	36.2	83.7
	+ *SALT*	**32.7**	**62.8**	**27.8**	**47.8**	**83.0**	**96.5**	**88.0**	**97.3**	**37.7**	**85.0**
	DAPO	29.0	61.3	26.0	45.0	82.2	95.0	84.8	96.8	36.0	80.9
	+ *SALT*	**32.9**	**63.8**	**28.0**	**47.0**	**84.1**	**96.8**	**88.0**	**98.0**	**38.5**	**82.9**
Deepseek-Distill-Qwen-7B											
∼	Vanilla	50.7	78.0	36.7	64.0	90.3	97.4	94.9	98.8	48.9	82.4
MATH-TRAIN	GRPO	60.7	77.3	42.0	71.3	89.9	97.7	94.9	99.0	46.8	83.8
	+ *SALT*	**64.7**	**80.4**	**45.3**	**74.5**	**91.9**	**98.7**	**95.2**	98.9	**51.2**	**87.9**
	DAPO	54.0	79.3	42.0	62.7	90.1	97.5	94.2	99.0	48.9	83.7
	+ *SALT*	**57.3**	**82.2**	**44.8**	**66.6**	**91.7**	**99.0**	**95.2**	**99.1**	**52.2**	**88.1**
DAPO-MATH	GRPO	61.5	78.0	42.8	72.1	90.4	97.8	95.0	99.1	47.6	84.6
	+ *SALT*	**65.4**	**81.2**	**46.5**	**75.5**	**92.4**	**99.3**	**96.4**	**99.2**	**51.3**	**89.2**
	DAPO	54.8	80.0	43.0	64.0	90.4	97.6	94.4	99.1	49.2	84.1
	+ *SALT*	**56.6**	**81.6**	**46.3**	**68.3**	**92.1**	**99.2**	**95.4**	**99.3**	**53.1**	**88.7**

Thus, the surrogate deviation is controlled by the reweighting magnitude and mini-batch gradient geometry. When the original GRPO-style update is weakened by signed cancellation, *SALT* adaptively increases the spectral-tail contribution through n_{eff}. This is a biased surrogate: we do not claim true-policy-gradient unbiasedness, global convergence, or monotonic policy improvement. The next experiments section evaluates whether cancellation reduction outweighs coefficient distortion in the measured settings, and Appendix F.5 quantifies the corresponding trade-off.

4 Experiment

In this section, we conduct extensive experiments to answer the following research questions:

• (**RQ1**) Does *SALT* consistently improve GRPO-style performance across benchmarks and model scales, and are the gains cost-effective given the added geometry module?

• (**RQ2**) How does *SALT* affect exploration and optimization dynamics compared with alternative exploration controls?

• (**RQ3**) Which components of *SALT* are necessary, and how do they contribute to performance and geometric metrics?

• (**RQ4**) Does *SALT* make larger rollout groups actually useful under a fixed rollout budget, and why?

• (**RQ5**) Does *SALT* generalize across RLVR objectives and task domains?

4.1 Experimental Setup

Data and Evaluation. We evaluate *SALT* on GPQA-Diamond [37] and four math reasoning benchmarks: AIME24 [20], AIME25 [30], GSM8K [7] and MATH500 [25]. These datasets span scientific QA and mathematical problems. To ensure the stability of experiments, we run each experiment five times and report the average results with Accuracy and Pass@8. For the original code setting, we train on MBPP [3] and evaluate on HumanEval/HumanEval+ [6, 27] in Appendix F.1; additional code and logic evaluations are reported in Section 4.5.

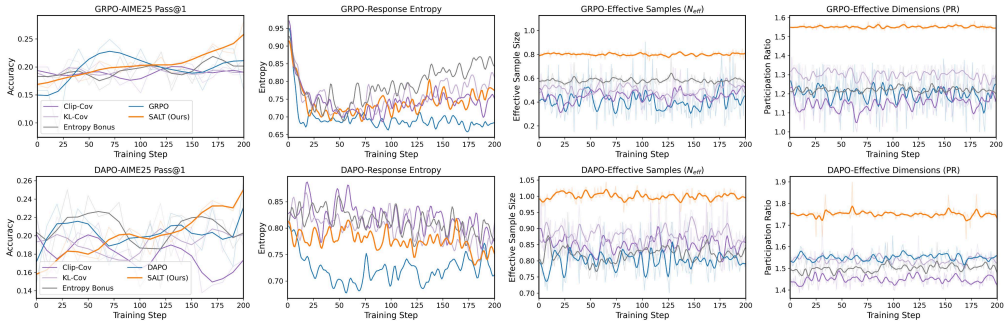

Figure 3: Training dynamics on MATH-TRAIN with GRPO (top) and DAPO (bottom), showing AIME25 accuracy, response entropy, PR, and n_{eff}. Entropy-based baselines increase response entropy but yield limited gains in geometry metrics and accuracy. In contrast, *SALT* improves both PR and n_{eff}, **achieving stronger performance via update exploration, not higher entropy.**

Training Dataset. We train our policy on two complementary math datasets for 2 epochs: MATH-Train-7.5k [18] and DAPO-Math-17k [54], aiming for a broader and more robust evaluation of training effectiveness across diverse math domains.

Settings. We use DeepSeek-Distill-Qwen [10] models at two scales (1.5B and 7B) as the base policy for training. We adopt a rule-based verifier to provide the reward signal. For each question we sample 8 responses with temperature 0.6 during training; we use the same sampling configuration for validation. Additional hyperparameters and implementation details are deferred to the Appendix C.

4.2 Main Result

(For RQ1-1) *SALT* consistently improves GRPO-style training. Table 2 compares GRPO/DAPO with *SALT* under the same training and sampling setup. Averaged over tasks, *SALT* improves the corresponding backbone by about +2.41 percentage points per setting, with paired 95% CIs in Table 13. Gains are largest on AIME24 and GPQA, with no consistent degradation. These results show that *SALT* robustly improves rollout aggregation, increasing both accuracy and pass@8 without extra exploration budget. Beyond the main setting, Appendix F.1 shows cross-model gains of +3 acc on average across additional model families, including Qwen3-8B [46] and LLaMA-3.1-8B [1], and further demonstrates that *SALT* improves acc on HumanEval [6] in execution-verifiable code RLVR.

Table 3: Average performance aggregated over reasoning benchmarks under GRPO and DAPO training. **Bold** denotes better performance.

Dataset	Method	Pass@1	Pass@8
MATH-TRAIN	GRPO	51.2	75.3
	+ Entropy Bonus	51.0	75.1
	+ Clip Cov	51.6	<u>75.8</u>
	+ KL Cov	<u>51.8</u>	<u>75.8</u>
	+ *SALT*	**53.6**	**78.1**
	DAPO	51.8	75.8
	+ Entropy Bonus	51.7	75.9
	+ Clip Cov	51.9	76.0
	+ KL Cov	<u>52.0</u>	<u>76.1</u>
	+ *SALT*	**53.1**	**76.9**
DAPO-MATH	GRPO	51.4	75.8
	+ Entropy Bonus	51.6	76.0
	+ Clip Cov	52.1	75.6
	+ KL Cov	<u>52.2</u>	<u>76.4</u>
	+ *SALT*	**53.8**	**77.9**
	DAPO	51.6	75.8
	+ Entropy Bonus	51.8	76.1
	+ Clip Cov	52.3	75.8
	+ KL Cov	<u>52.5</u>	<u>76.4</u>
	+ *SALT*	**54.3**	**77.7**

(For RQ1-2) *SALT* provides favorable cost–benefit. Since *SALT* adds a geometry module, we further evaluate whether its gains justify the extra wall-clock cost. Appendix G reports end-to-end training time. Across models and training sets, *SALT* adds about 7.7% average overhead in training time while improving both geometric effective metrics and accuracy by an average of 2.41 points. Thus, the controlled departure from GRPO in Section 3.3 is practically worthwhile, yielding measurable gains with modest extra cost.

(For RQ2) *SALT* promotes effective exploration beyond entropy-increasing controls. Figure 3 tracks AIME25 test performance together with response entropy, effective dimensionality PR

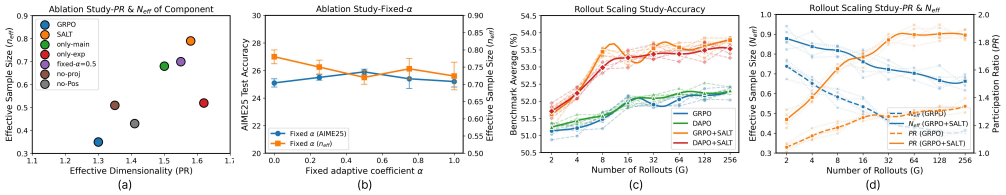

Figure 4: **Ablation and rollout-scaling analysis of *SALT*.** (a) *SALT* achieves the best PR–n_{eff} trade-off; ablations degrade update geometry. (b) Fixed-α exposes a brittle exploration–cancellation trade-off avoided by adaptive mixing. (c–d) With a fixed rollout budget, *SALT* better uses larger groups by raising PR while preserving n_{eff}, yielding consistent gains over GRPO/DAPO.

(Eq. (5)), and effective sample size n_{eff} (Eq. (6)) during GRPO training on both MATH-TRAIN and DAPO-MATH. Entropy-oriented baselines (entropy bonus, clip-cov, KL-cov [8, 43, 29]) increase entropy as intended, but their improvements in PR and n_{eff} remain limited and do not consistently translate into better AIME25 performance. In contrast, *SALT* achieves the strongest AIME25 results while simultaneously yielding higher PR and n_{eff}, indicating that *SALT* turns rollouts into more diverse and less-canceling update signals rather than merely amplifying sampling entropy.

Table 3 further confirms this trend across diverse benchmarks, where *SALT* consistently outperforms standard GRPO and entropy-promoting controls. In particular, improvement is observed on both the 1.5B and 7B scales (see Appendix F.2 for full results and additional analysis).

4.3 Ablation Study

(For RQ3) Both the dual-channel structure and cancellation-aware mixing are necessary. Figure 4(a) compares *SALT* with its ablations in the PR–n_{eff} plane. Removing either channel hurts update geometry: *only-main* reduces PR, while *only-exp* increases cancellation and lowers n_{eff}. Other controls show similar degradation: *no-proj/rand-proj* move toward the baseline, *no-Pos* reduces n_{eff}, and alternative mixing rules fail to reach the high-PR/high-n_{eff} regime. In contrast, full *SALT* achieves the best geometry and benchmark performance, indicating more diverse and less-canceling updates. Details are provided in Appendix F.3.1.

Figure 4 (b) sweeps a fixed mixing coefficient $\alpha \in \{0, 0.25, 0.5, 0.75, 1.0\}$ and exposes a clear trade-off: larger α boosts exploration but lowers n_{eff} via stronger gradient cancellation, while smaller α preserves n_{eff} but under-explores and hurts accuracy. This sensitivity makes fixed-α tuning brittle and *SALT* avoids it by adapting the mixing strength online. It further shows the realized adaptive α_t over training, illustrating that *SALT* replaces this brittle knob with an online-adjusted mixing signal.

Beyond these component ablations, Appendix F.3.2 further introduces a matched-coefficient subspace intervention, which preserves the coefficient distribution of *SALT* while breaking its alignment with the true mini-batch gradient geometry. The degradation under these matched null variants, together with consistently low clipping fractions, suggests that ***SALT*'s gains come from geometry-aligned coefficient assignment rather than coefficient redistribution or PPO clipping artifacts.**

4.4 Rollout Scaling Study

(For RQ4-1) *SALT* makes larger rollout groups useful under a fixed budget. We study whether *SALT* makes larger rollout groups G useful under a fixed rollout budget. Although increasing G provides more rollouts, it can also amplify gradient cancellation, so additional samples may not translate into stronger updates. As shown in Figure 4 (c), when G increases from 2 to 256, GRPO and DAPO exhibit diminishing returns, whereas *SALT* yields consistent gains across group sizes, with larger improvements at larger G. This indicates that *SALT* better utilizes additional rollouts under the same rollout budget, rather than merely benefiting from more compute.

(For RQ4-2) Why: *SALT* increases diversity while mitigating cancellation via adaptive mixing. We also track participation ratio (PR) for diversity and effective sample size n_{eff} for gradient alignment in Figure 4 (d). As G increases, PR grows for both methods, but baseline n_{eff} decreases, revealing stronger cancellation. SALT maintains higher n_{eff} while further improving PR, allowing larger G to translate into effective updates. Overall, *SALT* enables large-G scaling under a fixed

Table 4: **Generalization across RLVR objectives and task domains.** (a) Average Pass@1/Pass@8 over AIME24, AIME25, GSM8K, MATH-500, and GPQA using DeepSeek-Distill-Qwen-7B trained on MATH-TRAIN. Each objective is evaluated with and without *SALT* under a matched training budget. (b) Pass@1 on code and logic benchmarks; **Canc.** denotes the mean cancellation severity measured during training. Results are averaged over five seeds.

(a) Across RLVR objectives

Objective	Base	+*SALT*	Δ P@1
GRPO	66.9/85.8	**69.7/88.1**	**+2.8**
DAPO	65.8/84.4	**68.2/87.0**	**+2.4**
CISPO	67.4/86.2	**69.9/88.4**	**+2.5**
GSPO	67.8/86.5	**70.2/88.6**	**+2.4**
DPPO	68.1/86.9	**70.5/88.9**	**+2.4**

(b) Across task domains

Benchmark	GRPO	+*SALT*	Δ	Canc.
HumanEval	78.0	**79.3**	**+1.3**	0.21
HumanEval+	72.4	**73.5**	**+1.1**	0.21
LiveCodeBench-v5	24.6	**27.0**	**+2.4**	0.38
ZebraLogic	31.2	**34.6**	**+3.4**	0.52
K&K held-out	58.4	**62.1**	**+3.7**	0.55

budget by jointly increasing PR and maintaining n_{eff} through adaptive main/exploration mixing. Additional mechanism evidence is provided in Appendix F.4.

4.5 Generalization across Objectives and Domains

(For RQ5-1) *SALT* generalizes across RLVR objectives. Table 4(a) compares matched-budget training with GRPO, DAPO, CISPO [28], GSPO [58], and DPPO [35]. Adding *SALT* improves every objective by 2.4–2.8 Pass@1 points, indicating that its sample-geometry reweighting is complementary to different update constraints rather than specific to one clipping formulation.

(For RQ5-2) *SALT* generalizes beyond mathematical reasoning. Table 4(b) extends the evaluation to code and logic. The gain is modest when measured cancellation is mild on HumanEval/HumanEval+ [6, 27], increases on LiveCodeBench-v5 [21], and is largest on ZebraLogic [26] and K&K [48], where cancellation is strongest. Thus, the cross-domain gains track the diagnosed cancellation severity rather than requiring a uniform intervention across tasks.

5 Related Work

Reinforcement Learning in Reasoning. Recent progress in language model reasoning has increasingly relied on reinforcement learning (RL) to improve multi-step reasoning by directly optimizing task-level objectives on model-generated solutions [13]. A representative and widely used PPO-style [40] approach in this setting is *Group Relative Policy Optimization*, which constructs group-relative advantages from multiple completions per prompt without an explicit critic [42, 11].

Optimization of GRPO. Recent GRPO variants improve training stability by modifying the optimization objective: DAPO introduces decoupled clipping and dynamic sampling [54], while GSPO uses sequence-level ratios to stabilize RL for large Mixture-of-Experts models [58]. Other works modify update constraints, reweight advantages, or add entropy-aware advantages, self-correction, and multi-stage refinement to improve reasoning [28, 35, 33, 56, 57, 52, 24, 14, 60]. However, these methods mainly reshape gradient signals rather than **explicitly modeling gradient features**, leaving an important gap.

6 Conclusion

Limitations. *SALT* uses an LM-head gradient proxy and extra Gram computation, so its efficiency and geometry estimates may vary across architectures and training regimes. It is also a biased geometry-reweighted surrogate, whose gains depend on reducing signed cancellation enough to offset coefficient distortion; we do not claim true-policy-gradient unbiasedness, global convergence, or monotonic policy improvement.

Future work. Beyond the math and code RLVR settings studied here, future work will explore noisier learned rewards, multi-modal policies, agentic or search-based tasks, and other forms of group-based policy optimization. *Less redundancy, more progress.*

Acknowledgments and Disclosure of Funding

This work was conducted at Bilibili Inc. and was supported by Bilibili Inc. through personnel time, computing resources, and research infrastructure. The authors declare no competing interests beyond the affiliations and support disclosed in this paper.

References

[1] AI@Meta. Llama 3 model card, 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

[2] Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=ryghZJBKPS.

[3] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[4] Powei Chang, Jinpeng Zhang, Bowen Chen, Chenyu Wang, Chenlu Guo, Yixing Zhang, Yukang Gao, JianXiang Xiang, Yue Gao, Chaoqun Sun, et al. Spice: Submodular penalized information-conflict selection for efficient large language model training. *arXiv preprint arXiv:2601.23155*, 2026.

[5] Junhua Chen, Zixi Zhang, Hantao Zhong, and Rika Antonova. Group policy gradient, 2025. URL https://arxiv.org/abs/2510.03679.

[6] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. Introduces GSM8K.

[8] Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The entropy mechanism of reinforcement learning for reasoning language models, 2025. URL https://arxiv.org/abs/2505.22617.

[9] Runpeng Dai, Linfeng Song, Haolin Liu, Zhenwen Liang, Dian Yu, Haitao Mi, Zhaopeng Tu, Rui Liu, Tong Zheng, Hongtu Zhu, and Dong Yu. Cde: Curiosity-driven exploration for efficient reinforcement learning in large language models, 2025. URL https://arxiv.org/abs/2509.09675.

[10] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[11] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. doi: 10.48550/arXiv.2501.12948. URL https://arxiv.org/abs/2501.12948.

[12] Marco Del Giudice. Effective dimensionality: A tutorial. *Multivariate Behavioral Research*, 56(3):527–542, 2021. doi: 10.1080/00273171.2020.1743631. URL https://doi.org/10.1080/00273171.2020.1743631.

[13] Fei Ding, Baiqiao Wang, Zijian Zeng, and Youwei Wang. Multi-layer grpo: Enhancing reasoning and self-correction in large language models, 2025. URL https://arxiv.org/abs/2506.04746.

[14] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training, 2025. URL https://arxiv.org/abs/2505.10978.

[15] Hamza Giaffar, Camille Rullán Buxó, and Mikio Aoi. The effective number of shared dimensions between paired datasets. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 4249–4257. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/giaffar24a.html.

[16] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies, 2017. URL https://arxiv.org/abs/1702.08165.

[17] James Hazelden. Fast neural tangent kernel alignment, norm and effective rank via trace estimation. *arXiv:2511.10796*, 2025. doi: 10.48550/arXiv.2511.10796. URL https://www.arxiv.org/abs/2511.10796.

[18] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

[19] Zhenyu Hou, Pengfan Du, Yilin Niu, Zhengxiao Du, Aohan Zeng, Xiao Liu, Minlie Huang, Hongning Wang, Jie Tang, and Yuxiao Dong. Does rlhf scale? exploring the impacts from data, model, and method, 2024. URL https://arxiv.org/abs/2412.06000.

[20] Hugging Face H4. Huggingfaceh4/aime_2024. https://huggingface.co/datasets/HuggingFaceH4/aime_2024, 2024. Accessed: 2026-01-16.

[21] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

[22] Leslie Kish. *Survey Sampling*. John Wiley & Sons, 1965.

[23] Hynek Kydlíček and Hugging Face. Math-verify, 2026. URL https://github.com/huggingface/Math-Verify. GitHub repository. Accessed 2026-01-16.

[24] Lianrui Li, Dakuan Lu, Jiawei Shao, Chi Zhang, and Xuelong Li. Scrpo: From errors to insights, 2025. URL https://arxiv.org/abs/2511.06065.

[25] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[26] Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. ZebraLogic: On the scaling limits of llms for logical reasoning. *arXiv preprint arXiv:2502.01100*, 2025. doi: 10.48550/arXiv.2502.01100. URL https://arxiv.org/abs/2502.01100.

[27] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=1qvx610Cu7.

[28] MiniMax, Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025. doi: 10.48550/arXiv.2506.13585. URL https://arxiv.org/abs/2506.13585.

[29] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016. URL https://arxiv.org/abs/1602.01783.

[30] OpenCompass. opencompass/aime2025. https://huggingface.co/datasets/opencompass/AIME2025, 2025. Accessed: 2026-01-16.

[31] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

[32] Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, 2023. URL https://proceedings.mlr.press/v202/park23c.html.

[33] Daniil Plyusov, Alexey Gorbatovski, Boris Shaposhnikov, Viacheslav Sinii, Alexey Malakhov, Daria Korotyshova, and Daniil Gavrilov. F-grpo: Don't let your policy learn the obvious and forget the rare. *arXiv preprint arXiv:2602.06717*, 2026. doi: 10.48550/arXiv.2602.06717. URL https://arxiv.org/abs/2602.06717.

[34] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://arxiv.org/abs/2002.08484.

[35] Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. Rethinking the trust region in llm reinforcement learning. *arXiv preprint arXiv:2602.04879*, 2026. doi: 10.48550/arXiv.2602.04879. URL https://arxiv.org/abs/2602.04879.

[36] Stefano Recanatesi, Serena Bradde, Vijay Balasubramanian, Nicholas A. Steinmetz, and Eric Shea-Brown. A scale-dependent measure of system dimensionality. *Patterns*, 3(8):100555, 2022. doi: 10.1016/j.patter.2022.100555. URL https://www.sciencedirect.com/science/article/pii/S266638992200160X.

[37] 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, 2023.

[38] John Schulman. Approximating kl divergence. Blog post, 2016. URL http://joschu.net/blog/kl-approx.html. Accessed 2026-01-16. Some secondary sources cite the post as 2020.

[39] John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2016. URL https://arxiv.org/abs/1506.02438.

[40] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. doi: 10.48550/arXiv.1707.06347. URL https://arxiv.org/abs/1707.06347.

[41] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

[42] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300. URL https://arxiv.org/abs/2402.03300.

[43] Han Shen. On entropy control in llm-rl algorithms, 2025. URL https://arxiv.org/abs/2509.03493.

[44] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

[45] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2 edition, 2018.

[46] Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

[47] Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints, 2023. URL https://arxiv.org/abs/2309.16240.

[48] Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*, 2024. doi: 10.48550/arXiv.2410.23123. URL https://arxiv.org/abs/2410.23123.

[49] Yixuan Even Xu, Yash Savani, Fei Fang, and J. Zico Kolter. Not all rollouts are useful: Down-sampling rollouts in llm reinforcement learning, 2025. URL https://arxiv.org/abs/2504.13818.

[50] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

[51] Yaning Yang, Elaine F. Remmers, Chukwuma B. Ogunwole, Daniel L. Kastner, Peter K. Gregersen, and Wentian Li. Effective sample size: Quick estimation of the effect of related samples in genetic case-control association analyses. *arXiv: q-bio/0611093*, 2006. URL https://arxiv.org/abs/q-bio/0611093.

[52] Jian Yao, Ran Cheng, Xingyu Wu, Jibin Wu, and Kay Chen Tan. Diversity-aware policy optimization for large language model reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=5eZ0iykpDU. Spotlight.

[53] Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1998–2007. PMLR, 2018. URL https://proceedings.mlr.press/v84/yin18a.html.

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

[55] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning, 2020. URL https://arxiv.org/abs/2001.06782.

[56] Jixiao Zhang and Chunsheng Zuo. GRPO-LEAD: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5642–5654, 2025. URL https://aclanthology.org/2025.emnlp-main.287/.

[57] Xingjian Zhang, Siwei Wen, Wenjun Wu, and Lei Huang. Edge-grpo: Entropy-driven grpo with guided error correction for advantage diversity, 2025. URL https://arxiv.org/abs/2507.21848.

[58] Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025. doi: 10.48550/arXiv.2507.18071. URL https://arxiv.org/abs/2507.18071.

[59] Boris Zhestiankin and Maria Ponomareva. Zhestyatsky at SemEval-2021 task 2: ReLU over cosine similarity for BERT fine-tuning. In Alexis Palmer, Nathan Schneider, Natalie Schluter, Guy Emerson, Aurelie Herbelot, and Xiaodan Zhu, editors, *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 163–168, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.semeval-1.17. URL https://aclanthology.org/2021.semeval-1.17/.

[60] Yujun Zhou, Zhenwen Liang, Haolin Liu, Wenhao Yu, Kishan Panaganti, Linfeng Song, Dian Yu, Xiangliang Zhang, Haitao Mi, and Dong Yu. Evolving language models without labels: Majority drives selection, novelty promotes variation, 2025. URL https://arxiv.org/abs/2509.15194.

Appendix

A Proof of the residual-gradient norm bound

This appendix provides a detailed derivation of the norm bound (Eq. (11)) used in Section 2.3. The argument is deterministic, which holds for each prompt group without invoking expectations, and relies only on standard Euclidean inequalities.

Setup. Fix a prompt q. GRPO-style training samples G rollouts $\{o_{q,i}\}_{i=1}^{G}$ and forms the group-averaged score-gradient update

$$g_q(\theta) \triangleq \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i}\, s_{q,i}, \quad s_{q,i} \triangleq \nabla_\theta \log \pi_\theta(o_{q,i} \mid q). \tag{22}$$

In standard GRPO-style implementations, the advantage is centered within each prompt group [41], hence

$$\sum_{i=1}^{G} \hat{A}_{q,i} = 0. \tag{23}$$

Define the within-group mean score gradient and residuals:

$$\bar{s}_q \triangleq \frac{1}{G}\sum_{i=1}^{G} s_{q,i}, \qquad \delta_{q,i} \triangleq s_{q,i} - \bar{s}_q. \tag{24}$$

By construction, the residuals sum to zero:

$$\sum_{i=1}^{G} \delta_{q,i} = \sum_{i=1}^{G} s_{q,i} - G\bar{s}_q = G\bar{s}_q - G\bar{s}_q = 0. \tag{25}$$

Residual-only form of the group update. Substituting $s_{q,i} = \bar{s}_q + \delta_{q,i}$ into $g_q(\theta)$ yields

$$\begin{aligned} g_q(\theta) &= \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i}(\bar{s}_q + \delta_{q,i}) \\ &= \frac{1}{G}\Big(\sum_{i=1}^{G} \hat{A}_{q,i}\Big)\bar{s}_q + \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i}\delta_{q,i}. \end{aligned} \tag{26}$$

Using the group-centering property (Eq. (23)), the mean-component term vanishes, giving the residual-only identity

$$g_q(\theta) = \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i}\delta_{q,i}. \tag{27}$$

Norm bound via Cauchy–Schwarz. We now prove the bound (Eq. (11))

$$\|g_q(\theta)\|_2 \le \frac{1}{G}\Big(\sum_{i=1}^{G} \hat{A}_{q,i}^2\Big)^{1/2}\Big(\sum_{i=1}^{G} \|\delta_{q,i}\|_2^2\Big)^{1/2}. \tag{28}$$

Proof. Define

$$u \triangleq \sum_{i=1}^{G} \hat{A}_{q,i}\delta_{q,i} \in \mathbb{R}^d, \tag{29}$$

so that

$$g_q(\theta) = \frac{1}{G}u. \tag{30}$$

Hence $\|g_q(\theta)\|_2 = \frac{1}{G}\|u\|_2$. We bound $\|u\|_2$ by duality. For any vector $v \in \mathbb{R}^d$ with $\|v\|_2 = 1$,

$$v^\top u = v^\top \sum_{i=1}^{G} \hat{A}_{q,i}\delta_{q,i} = \sum_{i=1}^{G} \hat{A}_{q,i}\, v^\top \delta_{q,i}. \tag{31}$$

Applying the scalar Cauchy–Schwarz inequality to the sequences $\{\hat{A}_{q,i}\}_{i=1}^{G}$ and $\{v^\top \delta_{q,i}\}_{i=1}^{G}$ gives

$$|v^\top u| \le \Big(\sum_{i=1}^{G} \hat{A}_{q,i}^2\Big)^{1/2} \Big(\sum_{i=1}^{G} (v^\top \delta_{q,i})^2\Big)^{1/2}. \tag{32}$$

Next, since $\|v\|_2 = 1$, we have for each i,

$$(v^\top \delta_{q,i})^2 \le \|v\|_2^2\, \|\delta_{q,i}\|_2^2 = \|\delta_{q,i}\|_2^2, \tag{33}$$

which implies

$$\sum_{i=1}^{G} (v^\top \delta_{q,i})^2 \le \sum_{i=1}^{G} \|\delta_{q,i}\|_2^2. \tag{34}$$

Combining Eq. (32) and (34) yields, for all $\|v\|_2 = 1$,

$$|v^\top u| \le \Big(\sum_{i=1}^{G} \hat{A}_{q,i}^2\Big)^{1/2} \Big(\sum_{i=1}^{G} \|\delta_{q,i}\|_2^2\Big)^{1/2}. \tag{35}$$

Finally, using the dual characterization of the Euclidean norm, $\|u\|_2 = \sup_{\|v\|_2=1} v^\top u$ (equivalently, $\sup_{\|v\|_2=1} |v^\top u|$), we obtain

$$\|u\|_2 \le \Big(\sum_{i=1}^{G} \hat{A}_{q,i}^2\Big)^{1/2} \Big(\sum_{i=1}^{G} \|\delta_{q,i}\|_2^2\Big)^{1/2}. \tag{36}$$

Multiplying by $1/G$ and recalling Eq. (29) proves the bound (Eq. (28) and (11)). □

B Residual-gradient diagnostic

Section 2.3 gives a within-prompt mean–residual analysis of why group-centered updates can be weak. We validate this diagnostic mechanism across rollout group sizes. This diagnostic is distinct from *SALT*'s mini-batch dominant/spectral-tail decomposition, which is defined in Section 3.1.

Metrics. For each prompt group, let $s_{q,i}$ denote the per-rollout score-gradient feature and decompose it as

$$s_{q,i} = \bar{s}_q + \delta_{q,i}, \qquad \bar{s}_q = \frac{1}{G}\sum_{i=1}^{G} s_{q,i}.$$

We compute the total, shared, and residual gradient energies as

$$E_{\mathrm{tot}}(q) = \sum_{i=1}^{G} \|s_{q,i}\|_2^2, \quad E_{\mathrm{sh}}(q) = G\|\bar{s}_q\|_2^2, \quad E_{\delta}(q) = \sum_{i=1}^{G} \|s_{q,i} - \bar{s}_q\|_2^2.$$

The residual-energy ratio is then defined as

$$\rho_\delta(q) = \frac{E_\delta(q)}{E_{\mathrm{tot}}(q)}.$$

We also measure the realized group-update norm

$$\|g_q\|_2 = \left\| \frac{1}{G}\sum_{i=1}^{G} \hat{A}_{q,i} s_{q,i} \right\|_2$$

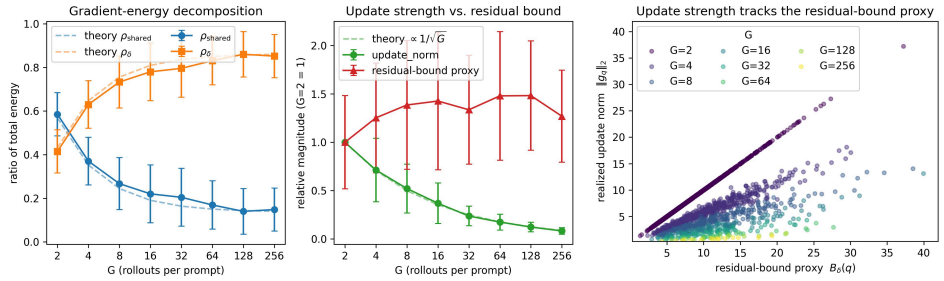

Figure 5: **Residual-gradient bottleneck under rollout scaling.** (a) As the rollout group size G increases, the shared-energy ratio decreases while the residual-energy ratio increases and saturates. (b) The residual-bound proxy remains bounded, whereas the realized update norm decreases with the group-averaging scale. (c) Across prompt groups, the realized update norm follows the residual-bound proxy, supporting that effective GRPO-style rollout scaling depends on non-redundant residual directions after group centering rather than on rollout count alone.

and compare it with the residual-energy bound

$$B_\delta(q) = \left(\frac{1}{G} \sum_{i=1}^{G} \|\delta_{q,i}\|_2^2 \right)^{1/2}.$$

In practice, we compute these diagnostic quantities from the same LM-head gradient-proxy features used by *SALT*, avoiding the need to materialize full-model per-sample gradients. They explain the group-centered cancellation mechanism but are not the projectors used by *SALT*. Equivalently, given the group-wise Gram matrix $K_q = S_q^\top S_q$, we compute

$$E_\delta(q) = \mathrm{tr}(K_q) - \frac{1}{G} \mathbf{1}^\top K_q \mathbf{1}, \quad \|g_q\|_2^2 = \frac{1}{G^2} \hat{A}_q^\top K_q \hat{A}_q.$$

Validation Results. Figure 5 validates the residual-bottleneck prediction. As G increases, the residual-energy ratio increases but saturates, while the residual-bound proxy remains bounded and the realized update norm decreases. Moreover, the update norm correlates with the residual-bound proxy both globally, with Pearson 0.56 and Spearman 0.52, and within each fixed G, where correlations remain above 0.95. Thus, larger rollout groups strengthen GRPO-style training only when they introduce non-redundant residual directions that survive group-centered aggregation.

C Implementation Details

In this section, we will provide more details on the implementation of our experiments in Section 4.1

C.1 Hyperparameter Configuration

For reproducibility, we implement our method in the VeRL [44] framework as a modular component, enabling plug-and-play integration with various GRPO-style algorithms. We conduct experiments using DeepSeek-Distill-Qwen models at two scales (1.5B and 7B) and all experiments are conducted on a single server equipped with 8x NVIDIA H20 GPUs. Table 5 summarizes the hyperparameter settings used in our experiments (including GRPO and DAPO). Additionally, in Section 4.2 we compare methods proposed by Cui et al. [8] for increasing policy entropy. By constraining updates to tokens with high covariance, these methods prevent entropy collapse (the specific formulas are provided in Appendix I).

Table 5: The hyperparameter configuration for training.

Hyperparameter	Value (default setting)
Training Setting	
Rollout samples per prompt	8
Rollout temperature	0.6
Learning rate	1e-6
Validation temperature	0.6
Entropy coefficient	0
KL loss coefficient	0.001
Prompt length	1024
Response length	8192
Train batch size	64
Mini-Batch Size	32
Training epochs	2
Additional Setting for DAPO	
Generation batch size	128
Filter groups enable	True
Max numbers of generation batches	10
Clip ratio (Low)	0.2
Clip ratio (High)	0.28
KL-Covariance	
KL covariance ratio	0.002
PPO KL coefficient	1
Clip ratio (Low)	0.2
Clip ratio (High)	0.2
Training epochs	2
Clip-Covariance	
Clip covariance ratio	0.0002
Clip covariance lb	1.0
Clip covariance ub	5.0
Clip ratio (Low)	1
Clip ratio (High)	1

C.2 System Prompt and Reward

System prompt. For all experiments, we applied a unified system prompt to constrain the model's output format, requiring explicit step-by-step reasoning and a clearly delineated final answer

> Please reason step by step, and put your final answer within \boxed{}.

which is the default prompt in VeRL [44].

Reward function. To better align the training signal with the reward for solving problems correctly, we use the Math-Verify library [23] to extract the final answer (see the reference code for details). For GRPO, the reward is set to 1 for a correct answer and 0 otherwise; for DAPO, the reward is 1 for a correct answer and -1 otherwise.

D Formal definitions of metrics

D.1 Metrics in redundancy observation

In Section 2.2.1, we introduce the metrics used to measure gradient redundancy. This section presents a detailed description of these metrics, which are divided into **in** and **out** categories.

In each iteration, we sample a mini-batch of prompts $\{q_b\}_{b=1}^{B}$. For each prompt q_b, we sample a group of G responses and obtain a set of gradient features $\{g_{b,i}\}_{i=1}^{G} \subset \mathbb{R}^d$. We use $g_{b,i}$ as the geometric object for measuring alignment and effective dimensionality. Throughout, we define the flattened index set of all sampled responses as $\mathcal{I} = \{(b,i) : b = 1,\dots,B,\ i = 1,\dots,G\}$ with $|\mathcal{I}| = BG$.

To quantify cancellation in the actual update, we additionally define the signed per-sample update contribution $u_{b,i} \in \mathbb{R}^{|\theta|}$ as the parameter-gradient contribution multiplied by the scalar coefficient used by the objective. Concretely, we instantiate

$$u_{b,i} \triangleq \hat{A}_{b,i}\, v_{b,i}, \tag{37}$$

where $v_{b,i}$ is the response-level parameter-gradient contribution and $\hat{A}_{b,i}$ is the normalized advantage.

(1) Positive cosine similarity. For any two vectors $x, y \in \mathbb{R}^d$, define the cosine similarity $\cos(x, y) \triangleq \langle x, y\rangle/(\|x\|_2\|y\|_2)$ and its positive version $[\cos(x, y)]_+ \triangleq \max\{0, \cos(x, y)\}$. For a set of m vectors $\{g_i\}_{i=1}^m$, we define the mean positive cosine similarity [59]

$$r(\{g_i\}_{i=1}^m) \triangleq \frac{2}{m(m-1)} \sum_{1\le i<j\le m} [\cos(g_i, g_j)]_+. \tag{38}$$

We report within-group and batch-level variants:

$$r_{\text{in}}(q_b) \triangleq r(\{g_{b,i}\}_{i=1}^G), \tag{39}$$

$$r_{\text{in,mean}} \triangleq \frac{1}{B}\sum_{b=1}^B r_{\text{in}}(q_b), \tag{40}$$

$$r_{\text{total}} \triangleq r(\{g_{b,i}\}_{(b,i)\in\mathcal{I}}). \tag{41}$$

(2) Participation ratio (PR). Given a set of m gradient features $\{g_i\}_{i=1}^m$, let $G = [g_1, \ldots, g_m] \in \mathbb{R}^{d\times m}$ and form the normalized Gram matrix

$$K \triangleq \frac{1}{m} G^\top G \in \mathbb{R}^{m\times m}. \tag{42}$$

Let $\{\lambda_i\}_{i=1}^m$ denote the eigenvalues of K (computed by a symmetric eigensolver); in implementation we clamp $\lambda_i \leftarrow \max\{0, \lambda_i\}$ for numerical stability. We define the participation ratio [36, 17]

$$\text{PR}(K) \triangleq \frac{(\sum_{i=1}^m \lambda_i)^2}{\sum_{i=1}^m \lambda_i^2}. \tag{43}$$

Larger $\text{PR}(K)$ indicates a less concentrated spectrum and thus a higher effective dimensionality of the set $\{g_i\}_{i=1}^m$.

We report within-group and batch-level variants:

$$\text{PR}_{\text{in}}(q_b) \triangleq \text{PR}(K_{\text{in}}(q_b)), \tag{44}$$

$$K_{\text{in}}(q_b) \triangleq \frac{1}{G} G_b^\top G_b, \;\; G_b \triangleq [g_{b,1}, \ldots, g_{b,G}], \tag{45}$$

$$\text{PR}_{\text{in,mean}} \triangleq \frac{1}{B}\sum_{b=1}^B \text{PR}_{\text{in}}(q_b), \tag{46}$$

$$\text{PR}_{\text{total}} \triangleq \text{PR}(K_{\text{total}}), \tag{47}$$

$$K_{\text{total}} \triangleq \frac{1}{BG} G_{\text{all}}^\top G_{\text{all}}, \tag{48}$$

where $G_{\text{all}} \triangleq [g_{b,i}]_{(b,i)\in\mathcal{I}}$ stacks all gradient features in the mini-batch.

(3) Effective sample size proxy. For a set of signed update vectors $\{u_i\}_{i=1}^m$, we define the effective sample size proxy

$$n_{\text{eff}}(\{u_i\}_{i=1}^m) \triangleq \frac{\|\sum_{i=1}^m u_i\|_2^2}{\sum_{i=1}^m \|u_i\|_2^2}. \tag{49}$$

This ratio increases when updates add coherently and decreases under cancellation [22, 51].

We report within- and cross-group variants:

$$n_{\text{eff}}^{\text{in}}(q_b) \triangleq n_{\text{eff}}(\{u_{b,i}\}_{i=1}^G), \tag{50}$$

$$n_{\text{eff,in,mean}} \triangleq \frac{1}{B}\sum_{b=1}^B n_{\text{eff}}^{\text{in}}(q_b), \tag{51}$$

and for cross-group aggregation we first compute prompt-level mean updates

$$\bar{u}_b \triangleq \frac{1}{G}\sum_{i=1}^G u_{b,i}, \tag{52}$$

then define

$$n_{\text{eff}}^{\text{out}} \triangleq n_{\text{eff}}(\{\bar{u}_b\}_{b=1}^B). \tag{53}$$

D.2 Metrics in evaluation

In Section 4.1, we evaluate the benchmarks with the following metrics:

Pass@k. *Pass@k* is a standard metric for math-generation benchmarks that measures the probability that at least one of k randomly selected candidate solutions solves the problem. Concretely, for each instance, we draw n candidate outputs (typically with stochastic decoding) and mark c of them as correct according to the evaluator (e.g., unit tests or an exact-match checker). The estimated pass@k is given by

$$\text{pass@}k = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}, \qquad 1 \le k \le n, \tag{54}$$

where $\binom{n-c}{k} = 0$ when $n - c < k$. This estimates the probability that a uniformly chosen subset of k samples contains at least one correct solution. Under the evaluation setting $n = 8$, Pass@1 equals $c/8$ and Pass@8 equals $\mathbf{1}\{c > 0\}$. We report $\text{pass@}k$ averaged over all instances and, to ensure the stability of the experiments, we will run validation five times and report the average results.

E Proxy Fidelity and Blockwise Whole-Model Geometry

SALT relies on the geometry of per-sample policy-gradient features, including sample-wise similarity, signed alignment, spectral concentration, and the dominant subspace used for the dominant/spectral-tail decomposition. Directly materializing full per-sample gradients over all model parameters is prohibitively expensive for LLM training. Therefore, in our practical implementation, we **use the LM-head/output-projection gradient as a lightweight proxy** for estimating the geometry required by *SALT*.

We emphasize that the proxy is not intended to match the full-model gradient entry-wise. ***SALT* only uses the proxy through low-dimensional sample-wise geometry**: the cosine-Gram structure, the signed alignment pattern relevant to cancellation, the participation-ratio spectrum, and the top eigenspace used to construct the dominant and spectral-tail channels. This appendix therefore evaluates **whether the LM-head proxy preserves these *SALT*-relevant geometric quantities against stronger references**, including full-gradient diagnostics on smaller models and blockwise whole-model sketches on larger models.

E.1 References and Protocol

For each validation checkpoint, we use the same rollouts, rewards, and group-normalized advantages as in training. We compare the LM-head proxy against three types of reference gradients.

Table 6: Proxy fidelity against full-gradient and blockwise references. Results are averaged over validation checkpoints and mini-batches. **The LM-head proxy preserves the geometry used by *SALT* substantially better than permutation and noise nulls.**

Baselines	Gram $\rho \uparrow$	Sign agree. $\uparrow$	PR rel. err. $\downarrow$	Subspace overlap $\uparrow$	k_t match $\uparrow$	a' cosine $\uparrow$
Reference Gradients Baselines						
Full gradient, small model	0.71 ± 0.06	0.82 ± 0.03	0.09 ± 0.04	0.78 ± 0.05	0.74 ± 0.07	0.86 ± 0.04
Blockwise aggregate	0.76 ± 0.05	0.84 ± 0.03	0.07 ± 0.03	0.81 ± 0.04	0.79 ± 0.06	0.88 ± 0.03
Early block	0.58 ± 0.07	0.75 ± 0.04	0.16 ± 0.06	0.64 ± 0.06	0.61 ± 0.08	0.77 ± 0.05
Middle block	0.64 ± 0.06	0.78 ± 0.04	0.13 ± 0.05	0.69 ± 0.05	0.66 ± 0.07	0.80 ± 0.05
Late block	0.73 ± 0.05	0.83 ± 0.03	0.08 ± 0.04	0.79 ± 0.05	0.77 ± 0.06	0.87 ± 0.04
Last block, disjoint	0.79 ± 0.04	0.85 ± 0.03	0.06 ± 0.03	0.84 ± 0.04	0.82 ± 0.05	0.90 ± 0.03
Random-block average	0.68 ± 0.06	0.80 ± 0.04	0.10 ± 0.04	0.73 ± 0.05	0.71 ± 0.07	0.84 ± 0.04
Null Baselines						
Permutation null	0.02 ± 0.04	0.51 ± 0.02	0.47 ± 0.15	0.18 ± 0.06	0.19 ± 0.08	0.08 ± 0.05
Noise null	0.00 ± 0.03	0.50 ± 0.02	0.56 ± 0.18	0.13 ± 0.05	0.12 ± 0.06	0.03 ± 0.04

Small-model full-gradient reference. First, on smaller models where diagnostic computation is tractable, we compute full trainable-parameter per-sample gradients. This directly tests whether the proxy reflects whole-model gradient geometry. These full gradients are used only for validation and are never materialized during *SALT* training.

Blockwise whole-model sketch. Second, for larger models, we compute blockwise per-sample gradients from representative Transformer blocks, including early, middle, late, and randomly sampled blocks. We also report an aggregated blockwise reference by combining the sampled-block Gram statistics. This blockwise sketch tests whether the LM-head proxy captures geometry beyond a single adjacent layer.

Disjoint last-block reference. Third, we retain the original disjoint last-block validation. The LM-head and last-block parameters have zero parameter overlap, so agreement cannot be explained by shared parameters.

Null baselines. We include two null baselines: a permutation null, which randomly permutes sample correspondence between proxy and reference gradients, and a noise null, which replaces proxy features with random vectors of matched dimension.

E.2 Metrics

Let p_i denote the LM-head proxy for sample i, and g_i a reference gradient. We evaluate only the geometric quantities used by *SALT*.

Gram correlation. We compute cosine-Gram matrices

$$\widetilde{K}^p_{ij} = \frac{\langle p_i, p_j \rangle}{\|p_i\|_2 \|p_j\|_2}, \quad \widetilde{K}^g_{ij} = \frac{\langle g_i, g_j \rangle}{\|g_i\|_2 \|g_j\|_2}, \tag{55}$$

and report Spearman correlation over off-diagonal entries.

Signed alignment. Since *SALT* targets signed cancellation, we report sign agreement between $\widetilde{K}^p_{ij}$ and $\widetilde{K}^g_{ij}$ on high-magnitude reference pairs.

Spectral consistency. We compare the participation ratio

$$\mathrm{PR}(K) = \frac{\left(\sum_r \lambda_r\right)^2}{\sum_r \lambda_r^2}, \tag{56}$$

and report PR error as well as the consistency of the selected dominant dimension k_t.

Subspace consistency. Let V^p_k and V^g_k be the top-k eigenspaces of the proxy and reference Gram matrices. We report principal-angle overlap, computed from the singular values of $(V^p_k)^\top V^g_k$.

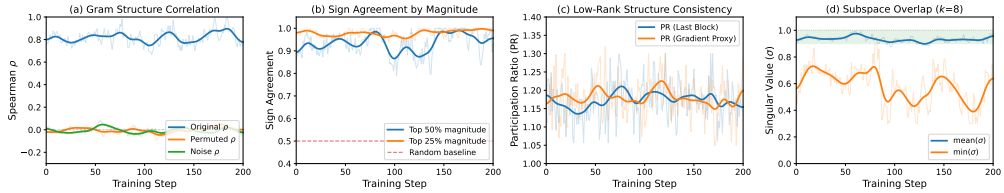

Figure 6: **Disjoint last-block proxy validation.** We validate the LM-head proxy against per-sample gradients from the last Transformer block on identical rollouts and advantages. Since the two parameter subsets are disjoint, their agreement cannot arise from shared parameters. We compare the geometry used by *SALT*: (a) cosine-Gram correlation is high while permutation/noise nulls are near zero; (b) high-magnitude signed pairs agree far above chance; (c) PR dynamics are closely tracked; and (d) the dominant top-k eigenspace (k=8) is well preserved, despite less stable tail directions.

Reweighting consistency. To measure whether the proxy leads to similar *SALT* decisions, we compare the reweighted advantages induced by proxy and reference geometries:

$$a'_{\text{proxy}} = P_k^p a + \alpha_t^p (I - P_k^p)[a]_+, \tag{57}$$

and analogously a'_{ref}. We report cosine similarity and rank correlation between them.

E.3 Results

Table 6 summarizes proxy fidelity across checkpoints and mini-batches. The LM-head proxy preserves the *SALT*-relevant geometry against both full-gradient diagnostics and blockwise references. It achieves substantially higher Gram correlation, signed agreement, PR consistency, subspace overlap, and reweighting consistency than permutation and noise nulls.

Figure 6 shows the original disjoint last-block validation. Despite zero parameter overlap, the LM-head proxy tracks the last-block reference in Gram structure, signed alignment, PR dynamics, and dominant subspace, while the null baselines remain near chance. Together with the full-gradient and blockwise results in Table 6, this supports using the LM-head proxy as a scalable estimator of the geometry required by *SALT*.

F Details of Experiment

F.1 Additional Experiments for RQ1

In Table 2, we show that adding *SALT* yields effective task-level gains. Figure 3 further illustrates the geometric characteristics observed during training, demonstrating that *SALT* can indeed bring meaningful improvements. In this section, we will focus on different model families and non-mathematical RLVR tasks to more intuitively show that this phenomenon is broadly prevalent in RLVR.

Cross-Models Generalization. We evaluate whether the signed low-rank structure and the resulting gradient cancellation observed in Figure persist across model families, and whether *SALT* consistently improves both gradient effectiveness (PR and n_{eff}) and downstream accuracy. For all models, we keep the RLVR pipeline identical to the main experiments. We match the total sampling budget and the number of optimization steps between GRPO and adding *SALT* [1, 46, 50, 41].

Table 7: Cross-model generalization under matched budgets. Across models using the same RLVR pipeline, GRPO+*SALT* consistently outperforms GRPO.

Model		Average	
		ACC	Pass@8
Deepseek-Distill-Qwen Model			
Deepseek-Distill-Qwen-1.5B	GRPO	51.2	75.3
	+ *SALT*	53.6	78.1
	+ Δ	+ 2.4	+ 2.8
Deepseek-Distill-Qwen-7B	GRPO	66.9	85.8
	+ *SALT*	69.7	88.1
	+ Δ	+ 2.8	+ 2.3
Instruct Model			
LLaMA-3.1-8B	GRPO	33.5	54.8
	+ *SALT*	36.1	57.0
	+ Δ	+ 2.6	+ 2.2
Qwen-3-8B	GRPO	80.1	89.0
	+ *SALT*	82.8	92.3
	+ Δ	+ 2.7	+ 3.3
Math Model			
Qwen-2.5-Math-1.5B	GRPO	43.7	62.5
	+ *SALT*	46.9	65.1
	+ Δ	+ 3.2	+ 2.6
Deepseek-Math-7B	GRPO	33.2	52.2
	+ *SALT*	37.2	55.7
	+ Δ	+ 4.0	+ 3.5

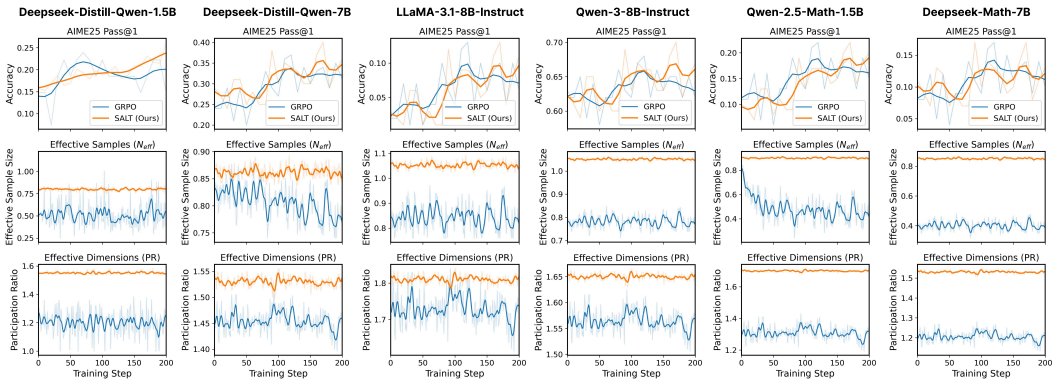

Figure 7: **Signed low-rank geometry persists across model families, and *SALT* mitigates gradient cancellation.** Training dynamics on AIME25 across multiple model families under matched sampling and optimization budgets. Top: Pass@1; Middle: effective sample size n_{eff}; Bottom: participation ratio (PR). Despite GRPO exhibiting low PR and low n_{eff} (strong cancellation), adding *SALT* consistently increases both PR and n_{eff}, translating into improved Pass@1.

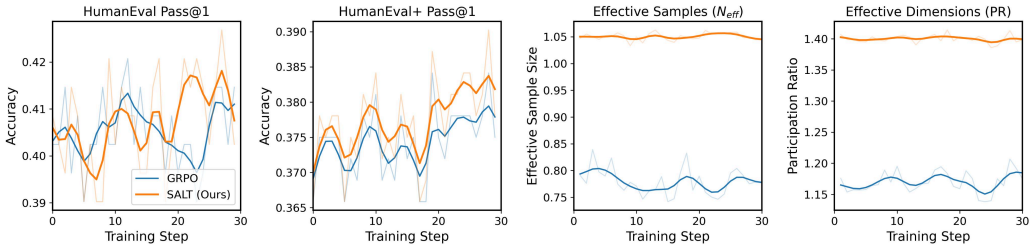

Figure 8: **Code RLVR results on MBPP with DeepSeek-R1-Distill-Qwen-7B.** GRPO exhibits strong cancellation (lower effective sample size n_{eff}) and an effectively low-rank update geometry (lower PR), while *SALT* consistently increases both n_{eff} and PR with higher Pass@1 performance.

Geometry persists across models. **We observe consistently low effective rank (small PR) and strong gradient cancellation (low n_{eff})** in Figure 7. *SALT* selectively fixes cancellation. Across models, *SALT* increases both PR and n_{eff}, indicating that rollouts are converted into effective update directions rather than being canceled by opposing alignments.

As summarized in Table 7, on average across all models, adding *SALT* improves ACC by **3.0** points and Pass@8 by **2.8** points compared to GRPO. ***SALT* consistently improves average accuracy and Pass@8** across distilled base models, instruction-tuned models, and math-specialized models, suggesting that *SALT*'s gains stem from mitigating gradient cancellation rather than merely encouraging randomness.

Code RLVR. We further evaluate *SALT* on code generation with execution-based verification, where rewards are obtained by running unit tests. This setting serves as an orthogonal, non-math RLVR domain with a high-precision verifier, enabling us to examine whether the signed low-rank geometry and the effectiveness of *SALT* generalize beyond mathematical reasoning.

To better expose this phenomenon, we train DeepSeek-R1-Distill-Qwen-7B for five epochs on the MBPP training set [3] with a batch size of 32. As shown in Figure 8, signed low-rank geometry persists during code RLVR training: compared with GRPO, *SALT* yields modest but consistent Pass@1 gains on both HumanEval [6] and HumanEval+ [27], while also improving n_{eff} and PR, indicating less cancellation and more diverse effective updates.

F.2 Detailed Results for RQ2

This appendix provides the complete results and supplementary analyzes for RQ2 in Section 4.2.

Table 8: Per-benchmark performance comparison aggregated under GRPO and DAPO training. Results are averaged over 5 runs. **Bold** numbers indicate best performance within each block. Overall, compared with entropy-control methods, ***SALT* not only introduce more effective directions at the geometric level, but also achieve the best task performance.**

Training	Method	AIME24		AIME25		GSM8K		MATH-500		GPQA	
		ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8
Deepseek-Distill-Qwen-1.5B											
MATH-T	GRPO	29.3	59.3	25.6	43.3	80.7	94.8	85.6	96.7	35.0	82.2
	+ Entropy Bonus	29.0	59.1	25.4	43.2	80.5	94.7	85.4	96.6	34.8	82.1
	+ Clip Cov	29.6	60.1	25.9	44.2	81.1	95.3	86.0	97.0	35.2	82.4
	+ KL Cov	29.8	60.0	26.0	44.0	81.3	95.1	86.2	97.1	35.5	82.6
	+ *SALT*	**32.1**	**62.9**	**27.1**	**47.2**	**83.4**	**96.5**	**87.2**	**98.4**	**38.4**	**85.5**
	DAPO	**31.1**	59.9	23.9	44.3	81.0	96.1	86.1	97.0	37.0	81.7
	+ Entropy Bonus	30.0	60.0	24.0	44.5	81.2	96.2	86.0	97.0	37.1	81.8
	+ Clip Cov	30.3	60.1	24.3	44.6	81.4	96.4	86.2	97.1	37.3	82.0
	+ KL Cov	30.4	60.2	24.4	44.7	81.5	96.5	86.3	**97.2**	37.4	82.0
	+ *SALT*	30.6	**60.6**	**25.6**	**46.3**	**82.5**	**97.1**	**87.1**	97.0	**39.6**	**83.5**
DAPO-M	GRPO	29.1	59.0	23.6	44.2	82.2	95.3	85.9	96.8	36.2	83.7
	+ Entropy Bonus	29.3	59.4	23.9	44.6	82.4	95.6	86.1	97.0	36.5	83.5
	+ Clip Cov	29.9	59.2	24.5	43.8	82.6	95.0	86.5	96.6	36.9	83.5
	+ KL Cov	30.0	59.8	24.6	45.0	82.6	95.8	86.6	97.2	37.0	84.2
	+ *SALT*	**32.7**	**62.8**	**27.8**	**47.8**	**83.0**	**96.5**	**88.0**	**97.3**	**37.7**	**85.0**
	DAPO	29.0	61.3	26.0	45.0	82.2	95.0	84.8	96.8	36.0	80.9
	+ Entropy Bonus	29.2	61.5	26.2	45.4	82.4	95.4	85.0	97.0	36.3	81.3
	+ Clip Cov	29.8	61.2	26.8	44.9	82.7	95.2	85.3	96.7	36.7	81.0
	+ KL Cov	30.0	61.8	27.0	45.8	82.9	95.6	85.6	97.2	37.0	81.6
	+ *SALT*	**32.9**	**63.8**	**28.0**	**47.0**	**84.1**	**96.8**	**88.0**	**98.0**	**38.5**	**82.9**
Deepseek-Distill-Qwen-7B											
MATH-T	GRPO	60.7	77.3	42.0	71.3	89.9	97.7	94.9	99.0	46.8	83.8
	+ Entropy Bonus	60.2	77.1	41.6	71.2	90.0	97.7	94.8	98.9	46.5	83.7
	+ Clip Cov	61.1	77.9	42.3	72.1	90.2	97.8	95.0	**99.1**	47.1	84.3
	+ KL Cov	61.5	78.1	42.6	72.0	90.1	97.8	95.1	99.0	47.3	84.4
	+ *SALT*	**64.7**	**80.4**	**45.3**	**74.5**	**91.9**	**98.7**	**95.2**	98.9	**51.2**	**87.9**
	DAPO	54.0	79.3	42.0	62.7	90.1	97.5	94.2	99.0	48.9	83.7
	+ Entropy Bonus	54.3	79.6	42.2	63.4	90.1	97.7	94.2	**99.2**	49.1	84.1
	+ Clip Cov	54.7	79.4	42.5	63.0	90.3	97.6	94.3	98.7	49.5	83.9
	+ KL Cov	54.9	79.8	42.7	63.7	90.2	97.7	94.3	99.1	49.6	84.4
	+ *SALT*	**57.3**	**82.2**	**44.8**	**66.6**	**91.7**	**99.0**	**95.2**	99.1	**52.2**	**88.1**
DAPO-M	GRPO	61.5	78.0	42.8	72.1	90.4	97.8	95.0	99.1	47.6	84.6
	+ Entropy Bonus	61.2	78.1	42.9	72.6	90.6	97.9	95.1	98.5	48.8	85.1
	+ Clip Cov	61.9	78.6	43.3	72.5	90.7	98.0	95.2	98.9	48.3	84.8
	+ KL Cov	62.2	78.8	43.4	73.2	90.8	98.0	95.2	98.8	48.5	85.6
	+ *SALT*	**65.4**	**81.2**	**46.5**	**75.5**	**92.4**	**99.3**	**96.4**	**99.2**	**51.3**	**89.2**
	DAPO	54.8	80.0	43.0	64.0	90.4	97.6	94.4	99.1	49.2	84.1
	+ Entropy Bonus	54.9	80.0	43.2	64.8	90.6	97.8	94.5	99.1	49.6	84.6
	+ Clip Cov	55.2	79.7	43.8	64.3	90.7	97.7	94.6	98.7	49.9	86.3
	+ KL Cov	55.0	80.2	44.3	65.4	90.7	97.9	94.6	99.1	50.4	84.8
	+ *SALT*	**56.6**	**81.6**	**46.3**	**68.3**	**92.1**	**99.2**	**95.4**	**99.3**	**53.1**	**88.7**

Full Results. Table 3 reports the averaged results for DeepSeek-Distill-Qwen-1.5B in the main paper. Here we provide the complete results per-benchmark for both the 1.5B and 7B models in Table 8.

Two patterns are consistent across model scales and benchmarks. First, ***SALT* achieves the strongest overall performance**, with gains often more pronounced in Pass@8 than in Pass@1, indicating improved effectiveness under multi-sample evaluation. Second, these gains are consistent with the mechanism emphasized throughout the paper: the update combines a stable main learning signal with an adaptive exploration component, which improves credit assignment and mitigates noisy or misaligned updates. As a result, *SALT* yields more reliable optimization under the same training budget, **translating into higher downstream accuracy rather than merely increasing or regulating response entropy**.

F.3 Additional Ablations for RQ3

F.3.1 Component Necessity Ablations

In Section 4.3, we conduct ablation studies on the individual components of our method *SALT*, and present further details below (Table 9).

Ablation definitions. We ablate *SALT* by selectively removing or modifying its key components while keeping the rest of the training pipeline identical. Let a denote the per-sample advantage vector, P_k the estimated common-gradient subspace projector, and $P_k^\perp = I - P_k$ its orthogonal complement. *SALT* forms the modified advantages as:

• ***SALT* (full).** $a' = P_k a + \alpha_t P_k^\perp [a]_+$, where α_t is the cancellation-aware mixing weight computed from effective sample size.

• ***only-main*.** $a' = P_k a$. This variant keeps only the dominant spectral channel and removes the spectral-tail channel.

• ***only-exp*.** $a' = \alpha_t P_k^\perp [a]_+$. This variant removes the dominant spectral channel and updates purely along the spectral-tail directions.

• ***no-Positive*.** $a' = P_k a + \alpha_t P_k^\perp a$. This variant removes positive gating in the spectral-tail channel, allowing signed tail coefficients to be reweighted.

• ***no-proj*.** $a' = a + \alpha_t [a]_+$. This variant removes the subspace decomposition entirely (i.e., no projection), retaining only the mixing and gating mechanism.

• ***rand-proj*.** $a' = \widetilde{P}_k a + \alpha_t \widetilde{P}_k^\perp [a]_+$, where $\widetilde{P}_k = \widetilde{V}_k \widetilde{V}_k^\top$ is constructed from a random orthonormal basis $\widetilde{V}_k \in \mathbb{R}^{m \times k_t}$ and $\widetilde{P}_k^\perp = I - \widetilde{P}_k$. This control removes data-driven subspace estimation while preserving the same two-channel structure.

To provide a more comprehensive assessment, we further evaluate alternative mixing schedules that control α_t using proxies other than n_{eff} (e.g., PR-driven or entropy-driven rules), while keeping all other settings fixed. These controls test whether *SALT*'s gains come from the cancellation-aware mixing, rather than from heuristic adaptivity.

• ***fixed-*α.** $a' = P_k a + \alpha P_k^\perp [a]_+$ with a constant $\alpha \in [0, 1]$ (shared across steps). This control removes cancellation-aware adaptivity and tests whether a fixed mixing weight suffices.

• ***PR-*α.** $a' = P_k a + \alpha_t P_k^\perp [a]_+$ with $\alpha_t = \mathrm{clip}\Big(\frac{PR_{\text{total}}}{m-1}, 0, 1\Big)$. This control drives mixing using effective dimensionality (PR) rather than cancellation (n_{eff}).

• ***Entropy-*α.** $a' = P_k a + \alpha_t P_k^\perp [a]_+$ with $\alpha_t = \mathrm{clip}\Big(\frac{H_t - H_{\min}}{H_{\max} - H_{\min}}, 0, 1\Big)$, where H_t is the policy entropy at step t and $H_{\min}, H_{\max}$ are computed over the same training run. This control uses exploration level as a proxy to set α_t instead of n_{eff}.

All variants use the same k_t and α_t computation when applicable, and differ only in how a is transformed into a'.

Benchmark performance. While the main text focuses on the PR–n_{eff} geometry in Figure 4 (a), we also report the *final benchmark performance* of *SALT* and all ablations under identical training budgets. Table 9 summarizes results across all benchmarks. Overall, the full *SALT* achieves the best performance and the most consistent gains. In contrast, removing either channel (*only-main*/*only-exp*), breaking the learned subspace structure (*no-proj*/*rand-proj*), or replacing cancellation-aware mixing with alternative schedules (*fixed-*α/*PR-*α/*Entropy-*α) leads to degraded performance, corroborating that **both the dual-channel design and cancellation-aware mixing are necessary**.

Visualization of α_t dynamics and clipping diagnostics. We further visualize the temporal evolution of the adaptive mixing coefficient α_t under different rollout group sizes. As shown in Figure 9, larger rollout groups consistently lead to higher α_t, indicating that *SALT* adaptively places more weight on the residual/exploration channel when redundancy and signed cancellation become stronger. This behavior is consistent with the design of the cancellation-aware mixing rule.

Table 9: *SALT* component ablations and alternative mixing schedules under GRPO (Deepseek-Distill-Qwen-1.5B; MATH-Train). Results are reported as Pass@1/Pass@8 on AIME24/25, GSM8K, MATH-500, and GPQA. **Bold** indicates the best performance. Overall, **the full *SALT* design achieves the strongest average results, and removing key components consistently degrades performance.**

Method	AIME24		AIME25		GSM8K		MATH-500		GPQA		Average	
	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8
Vanilla	29.1	59.3	22.0	42.6	80.3	95.0	85.5	96.4	34.5	82.4	50.3	75.1
GRPO	29.3	59.3	25.6	43.3	80.7	94.8	85.6	96.7	35.0	82.2	51.2	75.3
+ *only-main*	30.1	60.5	25.3	44.6	81.9	95.7	86.0	97.1	36.2	83.2	51.9	76.2
+ *only-exp*	29.8	60.0	25.0	44.0	81.5	95.4	85.9	97.0	35.8	82.9	51.6	75.9
+ *no-Pos*	29.6	59.7	24.7	43.8	81.1	95.2	85.8	96.8	35.4	82.5	51.3	75.6
+ *no-proj*	29.4	59.4	24.4	43.4	80.9	95.0	85.7	96.7	35.2	82.3	51.1	75.4
+ *rand-proj*	29.5	59.5	24.6	43.6	81.0	95.1	85.8	96.8	35.3	82.4	51.2	75.5
+ *PR-*α_t	30.3	61.0	25.4	45.0	82.0	95.9	86.1	97.3	36.5	83.7	52.1	76.6
+ *Ent-*α_t	30.0	60.6	25.2	44.5	81.8	95.6	86.0	97.2	36.1	83.3	51.8	76.2
+ *Fixed-*α_t	30.7	59.9	24.9	44.1	82.4	95.4	85.9	97.0	35.7	82.8	51.9	75.8
+ *SALT*	**32.1**	**62.9**	**27.1**	**47.2**	**83.4**	**96.5**	**87.2**	**98.4**	**38.4**	**85.5**	**53.6**	**78.1**

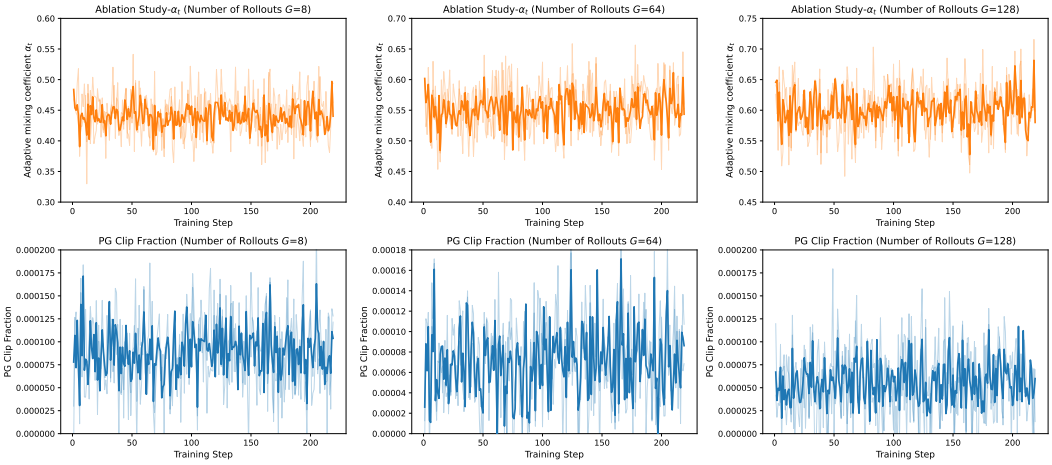

Figure 9: Adaptive mixing coefficient and policy-gradient clipping fraction under different rollout group sizes. As the rollout group size increases from $G = 8$ to $G = 128$, the adaptive mixing coefficient α_t increases, indicating that ***SALT* assigns more weight to the spectral-tail channel when larger rollout groups induce stronger redundancy and cancellation**. Meanwhile, the policy-gradient clipping fraction remains at the 10^{-4} level and stays below 2×10^{-4} throughout training, suggesting that the **observed gains are unlikely to be driven by PPO clipping artifacts**.

We also report the policy-gradient clipping fraction in the same figure to rule out clipping-induced artifacts. Across all rollout group sizes, the clipping fraction remains at the 10^{-4} level and stays below 2×10^{-4} throughout training. Therefore, the effect of *SALT* is unlikely to be driven by PPO clipping nonlinearities; instead, the diagnostic supports our interpretation that *SALT* improves update aggregation through geometry-aware residual reweighting.

F.3.2 Matched-Coefficient Subspace Intervention

The component ablations above verify the necessity of individual *SALT* components. However, modifying a component may also change the marginal distribution, scale, and positive mass of the reweighted coefficients, making it difficult to fully isolate whether the improvement comes from geometry-aware subspace alignment or from coefficient reshaping alone. To address this, we introduce a matched-coefficient subspace intervention that **preserves the coefficient distribution of *SALT* while breaking its alignment with the true mini-batch gradient geometry**.

Protocol. For each checkpoint and mini-batch, we fix the prompts, sampled rollouts, rewards, original group-normalized advantages, and the adaptive mixing coefficient. We first compute the

Table 10: **Matched-coefficient subspace intervention** on DeepSeek-Distill-Qwen-7B / MATH-TRAIN. **Coef. Dist.** indicates whether the marginal distribution of reweighted coefficients is matched to the corresponding standard *SALT* variant under the same backbone; **Geo. Align.** indicates whether the coefficient-to-sample assignment is aligned with the true data-dependent mini-batch gradient geometry. Matched null variants preserve the coefficient distribution of *SALT* while breaking this geometry alignment. Their degradation in effective update geometry indicates that ***SALT*'s gains come from geometry-aligned coefficient assignment rather than coefficient redistribution alone.**

Backbone	Method	Coef. Dist.	Geo. Align.	$D_a \downarrow$	$D_g \downarrow$	$n_{\text{eff}} \uparrow$	$U_\perp$ **Gain** $\uparrow$
GRPO	Backbone	–	–	0.000	0.000	0.31 ± 0.05	1.00×
GRPO	RandSubspace-*SALT*	No	No	0.22 ± 0.04	0.18 ± 0.03	0.50 ± 0.05	1.13×
GRPO	Permuted-*SALT*	Yes	No	0.24 ± 0.04	0.19 ± 0.03	0.43 ± 0.05	1.08×
GRPO	Matched-RandSubspace	Yes	No	0.24 ± 0.04	0.18 ± 0.03	0.55 ± 0.04	1.18×
GRPO	+ *SALT*	Yes	Yes	0.24 ± 0.04	0.16 ± 0.03	0.80 ± 0.04	1.47×
DAPO	Backbone	–	–	0.000	0.000	0.82 ± 0.03	1.00×
DAPO	RandSubspace-*SALT*	No	No	0.16 ± 0.03	0.13 ± 0.02	0.86 ± 0.04	1.07×
DAPO	Permuted-*SALT*	Yes	No	0.17 ± 0.03	0.13 ± 0.02	0.84 ± 0.04	1.04×
DAPO	Matched-RandSubspace	Yes	No	0.17 ± 0.03	0.12 ± 0.02	0.89 ± 0.03	1.10×
DAPO	+ *SALT*	Yes	Yes	0.17 ± 0.03	0.11 ± 0.02	1.00 ± 0.03	1.24×

standard *SALT* coefficients a'_{SALT} and record their sorted values. We then construct matched null variants by reassigning the same sorted coefficient values according to alternative sample orderings. This keeps the marginal distribution, norm, and positive mass of the coefficients identical to *SALT*, while disrupting their sample-wise alignment with the data-dependent gradient geometry.

Intervention variants. We consider two matched null variants. **Permuted-*SALT*** randomly permutes the assignment between a'_{SALT} and samples. **Matched-RandSubspace** first samples a random orthogonal subspace with the same dimension as the *SALT* dominant subspace, uses the induced residual scores to rank samples, and then assigns the sorted values of a'_{SALT} according to this random-subspace ranking. In both variants, the coefficient distribution is exactly matched to *SALT*; only the alignment between coefficients and the true mini-batch gradient geometry is broken.

Results and interpretation. As shown in Table 10, both matched null variants degrade the effective update geometry compared with *SALT*. Since these variants preserve the coefficient distribution, this degradation cannot be explained by changes in coefficient scale, sparsity, or positive mass. Instead, the result indicates that the gains of *SALT* rely on assigning reweighted coefficients to geometry-aligned samples, rather than merely changing the distribution of coefficient values. Together with the clipping diagnostic in Figure 9, where the policy-gradient clipping fraction remains below 2×10^{-4}, these results further suggest that the observed gains are unlikely to arise from PPO clipping artifacts.

F.4 Additional evidence for RQ4

In Section 4.4, we show that larger rollout groups G amplify both update diversity and gradient cancellation, and that *SALT* can better translate large-G rollouts into effective learning.

Here we provide additional analyses to connect these geometric trends to the underlying mechanism and to rule out simpler alternatives. First, we examine how *SALT* allocates updates between its main and exploration channels (Figure 4). Consistently, the exploration channel's contribution and the mixing coefficient α_t increase with G, indicating on-demand exploration under severe cancellation. Second, we compare against an entropy-only variant: while an entropy bonus improves randomness, it does not recover N_{eff} as effectively, leading to weaker or less reliable gains. Together, **these results suggest that increasing entropy alone is insufficient; cancellation-aware mixing is key.**

F.5 Bias–Cancellation Trade-off Diagnostics

Section 3.3 shows that *SALT* intentionally departs from the original GRPO-style estimator by replacing the group-normalized coefficients a with geometry-reweighted coefficients a'. This appendix quantifies whether this biased surrogate is both *controlled* and *useful*: controlled if the update deviation from GRPO remains moderate, and useful if the deviation increases the effective residual update and improves downstream accuracy.

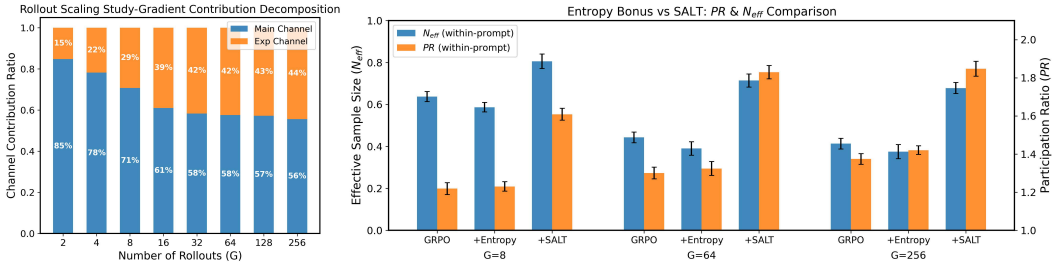

Figure 10: **Rollout scaling diagnostics:** (a) Decomposition of *SALT*'s gradient/update contributions into the main and exploration channels as the rollout group size G increases; the exploration share (and thus mixing) grows with G, **indicating on-demand exploration under stronger cancellation.** (b) Comparison with an entropy-only baseline at $G=8$, $G=64$ and $G=256$: **entropy increases randomness (higher PR) but does not recover N_{eff}** as effectively as *SALT*, leading to weaker or less reliable gains.

Table 11: Bias–cancellation trade-off diagnostics averaged over the last 20% training checkpoints and 5 seeds on DeepSeek-Distill-Qwen-7B / MATH-TRAIN. D_a measures coefficient distortion, D_g measures update deviation from the corresponding backbone, n_{eff} measures cancellation-adjusted effective update size, and $U_\perp$ measures spectral-tail update strength. ***SALT* introduces a moderate deviation while increasing effective update strength and downstream accuracy**.

Backbone	**Method**	$D_a \downarrow$	$D_g \downarrow$	$n_{\text{eff}} \uparrow$	$U_\perp$ **Gain** $\uparrow$	**Avg. ACC** $\uparrow$
GRPO	Backbone	0.000	0.000	0.31 ± 0.05	$1.00\times$	66.9
GRPO	+ *SALT*	0.24 ± 0.04	0.16 ± 0.03	0.80 ± 0.04	$1.47\times$	69.7
DAPO	Backbone	0.000	0.000	0.82 ± 0.03	$1.00\times$	65.8
DAPO	+ *SALT*	0.17 ± 0.03	0.11 ± 0.02	1.00 ± 0.03	$1.24\times$	68.2

Metrics. Let $S = [s_1, \ldots, s_m]$ denote the score-gradient feature matrix and $K = \frac{1}{m} S^\top S$ be the mini-batch Gram matrix. For any coefficient vector c, define the induced update

$$\hat{g}(c) = \frac{1}{m} Sc, \qquad \|\hat{g}(c)\|_2 = \sqrt{\frac{1}{m} c^\top K c}. \tag{58}$$

We measure the coefficient distortion and update deviation introduced by *SALT* as

$$D_a = \frac{\|a' - a\|_2}{\|a\|_2 + \epsilon}, \qquad D_g = \frac{\|\hat{g}(a') - \hat{g}(a)\|_2}{\|\hat{g}(a)\|_2 + \epsilon}. \tag{59}$$

To measure whether the deviation is useful, we report the cancellation-adjusted effective update size

$$n_{\text{eff}}(c) = \frac{c^\top K c}{\sum_{i=1}^m c_i^2 K_{ii} + \epsilon}, \tag{60}$$

and the spectral-tail update strength

$$U_\perp(c) = \left\| \frac{1}{m} S P_k^\perp c \right\|_2 = \sqrt{\frac{1}{m} (P_k^\perp c)^\top K (P_k^\perp c)}. \tag{61}$$

All quantities are computed from the same Gram statistics already used by *SALT*, without materializing full-model gradients.

Results. Table 11 reports averages over training checkpoints and seeds. The original GRPO/DAPO updates have zero coefficient distortion by definition. *SALT* introduces a nonzero but bounded deviation from the backbone update, while consistently increasing n_{eff}, spectral-tail update strength, and downstream accuracy. In the measured settings, this indicates that reducing signed cancellation outweighs the coefficient distortion; it is not a claim of true-policy-gradient unbiasedness or a universal improvement guarantee.

G Computational Cost

In this section, we will report the the complexity analysis and the wall-clock time cost.

G.1 Computational Complexity

Let B be the number of prompts in a mini-batch, G the number of rollouts per prompt, and $m \triangleq BG$ the total number of sampled responses in the batch. Our dual-channel reweighting introduces a *geometry module* on top of the base GRPO/DAPO [42, 54] update. Its cost depends on m (batch rollout count), rather than the full parameter size of the language model.

(1) Per-sample gradient feature extraction. For each response (b, i), we compute a last-layer gradient embedding proxy $g_{b,i}$ based on the output projection gradients (Eq. (19)). This step reuses quantities already produced in backprop (token-level $\nabla_z \ell$ and hidden states h), and aggregates over tokens to obtain a response-level feature. Denoting the average response length by T and the proxy feature dimension by d_g. In practice this overhead is typically small compared to the full forward+backward pass through the transformer stack [2, 34, 32].

Table 12: End-to-end wall-clock training time (Hour:Min) for GRPO/DAPO and variants, with and without *SALT*, on MATH-TRAIN/DAPO-MATH. All runs use the same training configuration; times include the full training loop.

Method	Time Cost (Hour:Min)	
	MATH-TRAIN	DAPO-MATH
Deepseek-Distill-Qwen-1.5B		
GRPO	07:21	15:41
+ *SALT*	07:58	17:08
+ Entropy Bonus	08:48	17:14
+ Clip Cov	07:53	16:03
+ KL Cov	07:34	16:01
DAPO	07:17	16:01
+ *SALT*	08:05	17:58
Deepseek-Distill-Qwen-7B		
GRPO	24:07	61:26
+ *SALT*	26:22	66:08
+ Entropy Bonus	25:52	17:14
+ Clip Cov	25:31	63:30
+ KL Cov	25:32	63:35
DAPO	24:31	62:23
+ *SALT*	25:58	66:04

(2) Forming the Gram matrix. Stack all features into $G_{\text{all}} \in \mathbb{R}^{d_g \times m}$ and form

$$K_{\text{total}} = \frac{1}{m} G_{\text{all}}^{\top} G_{\text{all}} \in \mathbb{R}^{m \times m}. \tag{62}$$

The memory to store K_{total} is $O(m^2)$. The time to compute $G_{\text{all}}^{\top} G_{\text{all}}$ is $O(d_g m^2)$; since d_g is fixed for a given proxy choice, the dominant scaling with m is $O(m^2)$ up to the constant factor d_g.

(3) Computing the dominant subspace (top-k_t eigenvectors). We only need the leading k_t eigenvectors to build the projector P_k. Using partial eigensolvers, the time complexity scales as

$$O(m^2 k_t), \tag{63}$$

and the additional memory beyond storing K_{total} is typically $O(mk_t)$ to hold the eigenvectors $V_k \in \mathbb{R}^{m \times k_t}$. The dominant dimension is set adaptively by $k_t = \text{clip}(\lfloor \text{PR}_{\text{total}} \rceil, 1, BG)$, so $k_t \ll m$ in our settings.

(4) Advantage projection and reweighting. Given V_k, the main/exploration channels are

$$a_{\text{main}} = P_k a, \qquad a_{\text{exp}} = P_k^{\perp} [a]_+, \tag{64}$$

where $P_k = V_k V_k^{\top}$ and $P_k^{\perp} = I - P_k$. We never need to materialize $P_k \in \mathbb{R}^{m \times m}$. Compute $P_k a$ as $V_k (V_k^{\top} a)$, which costs $O(mk_t)$; similarly for $P_k^{\perp}[a]_+$. Thus, projection/reweighting is linear in m up to k_t and is negligible compared with the Gram/eigensolver steps.

Overall overhead. The geometry module is dominated by (i) storing the Gram matrix ($O(m^2)$ memory) and (ii) extracting the top-k_t subspace ($O(m^2 k_t)$ time). Crucially, this overhead depends on $m = BG$ and is independent of the total parameter count of the backbone model, since we use last-layer gradient embeddings as a scalable proxy.

Table 13: Paired improvements of *SALT* over the corresponding baseline, reported as Δ_{-l}^{+u} (mean paired gain with 95% bootstrap CI over $N = 5$ matched seeds).

Training		AIME24		AIME25		GSM8K		MATH-500		GPQA	
		ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8	ACC	Pass@8
Deepseek-Distill-Qwen-1.5B											
MATH-T	GRPO	$+2.8^{+0.7}_{-0.6}$	$+3.6^{+1.0}_{-0.8}$	$+1.5^{+0.4}_{-0.3}$	$+3.9^{+0.9}_{-0.8}$	$+2.7^{+0.8}_{-0.6}$	$+1.7^{+0.6}_{-0.5}$	$+1.6^{+0.6}_{-0.4}$	$+1.7^{+0.5}_{-0.5}$	$+3.4^{+0.9}_{-0.8}$	$+3.3^{+0.8}_{-0.7}$
	DAPO	$-0.5^{+0.4}_{-0.3}$	$+0.7^{+0.6}_{-0.5}$	$+1.7^{+0.5}_{-0.4}$	$+2.0^{+0.7}_{-0.6}$	$+1.5^{+0.6}_{-0.5}$	$+1.0^{+0.3}_{-0.2}$	$+1.0^{+0.4}_{-0.3}$	$+0.0^{+0.9}_{-0.8}$	$+2.6^{+0.9}_{-0.7}$	$+1.8^{+0.6}_{-0.5}$
DAPO-M	GRPO	$+3.6^{+1.1}_{-0.9}$	$+3.8^{+1.0}_{-0.8}$	$+4.2^{+1.1}_{-0.9}$	$+3.6^{+1.0}_{-0.8}$	$+0.8^{+0.6}_{-0.8}$	$+1.2^{+0.5}_{-0.4}$	$+2.1^{+0.8}_{-0.6}$	$+0.5^{+0.6}_{-0.5}$	$+1.5^{+0.5}_{-0.4}$	$+1.3^{+0.7}_{-0.8}$
	DAPO	$+3.9^{+1.1}_{-0.9}$	$+2.5^{+0.9}_{-0.8}$	$+2.0^{+0.7}_{-0.6}$	$+2.0^{+0.8}_{-0.7}$	$+1.9^{+0.7}_{-0.6}$	$+1.8^{+0.7}_{-0.6}$	$+3.2^{+0.9}_{-0.8}$	$+1.2^{+0.5}_{-0.5}$	$+2.5^{+0.9}_{-0.7}$	$+2.0^{+0.7}_{-0.8}$
Deepseek-Distill-Qwen-7B											
MATH-T	GRPO	$+4.0^{+1.0}_{-0.8}$	$+3.1^{+0.9}_{-0.7}$	$+3.3^{+0.9}_{-0.8}$	$+3.2^{+1.0}_{-0.8}$	$+2.0^{+0.7}_{-0.6}$	$+1.0^{+0.3}_{-0.2}$	$+0.3^{+0.4}_{-0.3}$	$-0.1^{+0.3}_{-0.2}$	$+4.4^{+1.1}_{-0.9}$	$+4.1^{+1.1}_{-0.9}$
	DAPO	$+3.3^{+1.0}_{-0.8}$	$+2.9^{+0.9}_{-0.7}$	$+2.8^{+1.0}_{-0.8}$	$+3.9^{+1.1}_{-0.9}$	$+1.6^{+0.7}_{-0.6}$	$+1.5^{+0.6}_{-0.5}$	$+1.0^{+0.3}_{-0.2}$	$+0.1^{+0.3}_{-0.2}$	$+3.3^{+1.0}_{-0.8}$	$+4.4^{+1.2}_{-0.9}$
DAPO-M	GRPO	$+3.9^{+1.0}_{-0.9}$	$+3.2^{+0.9}_{-0.7}$	$+3.7^{+1.0}_{-0.8}$	$+3.4^{+1.0}_{-0.8}$	$+2.0^{+0.7}_{-0.6}$	$+1.5^{+0.6}_{-0.5}$	$+1.4^{+0.4}_{-0.3}$	$+0.1^{+0.4}_{-0.3}$	$+3.7^{+1.1}_{-0.9}$	$+4.6^{+1.2}_{-0.9}$
	DAPO	$+1.8^{+0.6}_{-0.6}$	$+1.6^{+0.6}_{-0.7}$	$+3.3^{+1.0}_{-0.9}$	$+4.3^{+1.2}_{-0.9}$	$+1.7^{+0.7}_{-0.6}$	$+1.6^{+0.6}_{-0.5}$	$+1.0^{+0.3}_{-0.2}$	$+0.2^{+0.4}_{-0.3}$	$+3.9^{+1.1}_{-0.9}$	$+4.6^{+1.2}_{-0.9}$

G.2 Time Cost

Table 12 reports the end-to-end wall-clock training time for each method under the same hardware and training setup. All runs use identical hyperparameters and differ only in the reweighting/regularization module. *SALT* adds a lightweight geometry module on top of GRPO/DAPO; consistent with the analysis in Appendix G.1, its overhead is small relative to the full forward/backward pass and mainly comes from forming the $m \times m$ Gram statistics.

H Statistical significance

For each setting, we run *SALT* and the corresponding baseline (GRPO/DAPO) with $N = 5$ matched seeds $\{2025, 2026, 2027, 2028, 2029\}$ under identical training and evaluation configurations. For each seed s, we define the paired improvement as $d_s \triangleq \text{score}_s^{\text{SALT}} - \text{score}_s^{\text{base}}$, and report the mean improvement $\Delta \triangleq \frac{1}{N}\sum_{s=1}^{N} d_s$.

We compute 95% confidence intervals via paired bootstrap: we resample $\{d_s\}_{s=1}^{N}$ with replacement for B times and take the 2.5/97.5 percentiles of the bootstrap means as $(\Delta_{\text{low}}, \Delta_{\text{high}})$. We report results in the compact form Δ_{-l}^{+u} where $u = \Delta_{\text{high}} - \Delta$ and $l = \Delta - \Delta_{\text{low}}$ in Table 13.

I Formulations of GRPO-style Variants

In Section 2.1, we briefly introduced the formulations of RL and GRPO. In this section, we will proceed from PPO to GRPO, and then present the formal formulations for several optimized variants [41].

Proximal Policy Optimization (PPO) Proximal Policy Optimization (PPO) [40] is a widely adopted actor–critic reinforcement learning algorithm, and it is frequently employed in the RL-based fine-tuning of large language models (LLMs). Specifically, PPO updates the LLM by maximizing the surrogate objective defined as follows:

$$\mathcal{J}_{\text{PPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\, o\sim\pi_{\theta_{\text{old}}}(O|q)} \frac{1}{|o|}\sum_{t=1}^{|o|} \min\left(\frac{\pi_\theta(o_t \mid q, o_{<t})}{\pi_{\theta_{\text{old}}}(o_t \mid q, o_{<t})}\hat{A}_t,\right.$$
$$\left.\text{clip}\left(\frac{\pi_\theta(o_t \mid q, o_{<t})}{\pi_{\theta_{\text{old}}}(o_t \mid q, o_{<t})}, 1-\epsilon,\, 1+\epsilon\right)\hat{A}_t\right)\Bigg]. \tag{65}$$

In this objective, $q \sim P(Q)$ denotes a query sampled from the data distribution, $o = (o_1, \ldots, o_{|o|})$ is a response generated by the behavior policy $\pi_{\theta_{\text{old}}}$ and $\hat{A}_t$ is an estimator of the advantage at time

step t. Given the value function V and the reward function R, $\hat{A}_t$ is computed using the Generalized Advantage Estimation (GAE) [39].

Group Relative Policy Optimization (GRPO) In PPO, the value function is often realized as an additional model comparable in size to the policy, leading to substantial memory and computation overhead. In the LLM setting, rewards are typically provided only at the sequence level, which further complicates learning token-level value estimates. To address this, Shao et al. [41] propose *Group Relative Policy Optimization* (GRPO), which eliminates the value function and instead uses the average reward of multiple samples for the same question as a baseline. Concretely, for each question q, GRPO draws $\{o_1, \ldots, o_G\} \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$ and maximizes the following objective (Eq. (3)):

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\, \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(O|q)} \left[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left(\min\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}\, \hat{A}_{i,t}, \right.\right.\right. \\ \left.\left.\left. \text{clip}\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}, 1-\epsilon,\ 1+\epsilon \right) \hat{A}_{i,t} \right) - \beta\, \mathbb{D}_{\text{KL}}[\pi_\theta \,\|\, \pi_{\text{ref}}] \right) \right]. \tag{66}$$

Rewards within the group are normalized to obtain a response-level advantage (Eq. (2))

$$\hat{A}_i = \frac{r_i - \text{mean}(r_1, r_2, \cdots, r_G)}{\text{std}(r_1, r_2, \cdots, r_G)}. \tag{67}$$

And different from the KL penalty term used in Eq. (65), the KL divergence is computed with the following unbiased estimator

$$\mathbb{D}_{\text{KL}}[\pi_\theta \,\|\, \pi_{\text{ref}}] = \frac{\pi_{\text{ref}}(o_{i,t} \mid q,\, o_{i,<t})}{\pi_\theta(o_{i,t} \mid q,\, o_{i,<t})} - \log\left(\frac{\pi_{\text{ref}}(o_{i,t} \mid q,\, o_{i,<t})}{\pi_\theta(o_{i,t} \mid q,\, o_{i,<t})} \right) - 1. \tag{68}$$

which is guaranteed to be positive [38].

Decoupled Clip and Dynamic Sampling Policy Optimization (DAPO) Yu et al. [54] propose the Decouple Clip and Dynamic Sampling Policy Optimization (DAPO) algorithm. DAPO samples a group of outputs $\{o_i\}_{i=1}^{G}$, and optimizes the policy via the following objective:

$$\mathcal{J}_{\text{DAPO}}(\theta) = \mathbb{E}_{(q,a)\sim\mathcal{D},\, \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \left[\frac{1}{\sum_{i=1}^{G}|o_i|} \sum_{i=1}^{G}\sum_{t=1}^{|o_i|} \min\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}\, \hat{A}_{i,t}, \right.\right. \\ \left.\left. \text{clip}\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})},\, 1-\epsilon_{\text{low}},\, 1+\epsilon_{\text{high}} \right) \hat{A}_{i,t} \right) \right], \\ \text{s.t. } 0 < \left| \{\, o_i \mid \text{is_equivalent}(a, o_i) \,\} \right| < G. \tag{69}$$

where the reward $r_{i,t}(\theta)$ and advantage $\hat{A}_{i,t}$ are the same with Eq. (3) in GRPO.

Entropy Bonus. We consider a standard entropy regularization baseline by augmenting the GRPO objective with a token-level policy entropy term. For a state $s_{i,t} = (q, o_{i,<t})$, define [43, 29]

$$\mathcal{H}_\theta(s_{i,t}) \triangleq -\sum_{a\in\mathcal{V}} \pi_\theta(a \mid s_{i,t}) \log \pi_\theta(a \mid s_{i,t}), \tag{70}$$

where $\mathcal{V}$ is the vocabulary. The entropy-bonus variant optimizes with entropy coefficient $\lambda_{\text{ENT}} \geq 0$.

$$\mathcal{J}_{\text{ENT}}(\theta) = \mathcal{J}_{\text{GRPO}}(\theta) + \lambda_{\text{ENT}} \cdot \mathbb{E}_{q,\{o_i\}_{i=1}^{G}} \left[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \mathcal{H}_\theta(q, o_{i,<t}) \right], \tag{71}$$

J Algorithm

Algorithm 1 provides the full per-step implementation of the *SALT* advantage reweighting procedure described in Section 3.

Algorithm 1 ***SALT***: Dual-Channel Advantage Reweighting for GRPO-style RLVR

Require: Policy π_θ; prompts $\{q_b\}_{b=1}^{B}$; rollouts per prompt G; small numerical constant ϵ.
Ensure: Updated parameters θ.

1: **Rollout and reward.** For each prompt q_b, sample G responses $\{o_{b,i}\}_{i=1}^{G} \sim \pi_\theta(\cdot|q_b)$ and compute rewards $\{r_{b,i}\}_{i=1}^{G}$.

2: **Group-normalized advantages.** Compute GRPO-style normalized advantages $\hat{A}_{b,i}$ within each prompt group and stack them into

$$a \leftarrow \mathrm{vec}(\hat{A}_{b,i}) \in \mathbb{R}^m, \qquad m = BG.$$

3: **Output-projection gradient features.** For each response (b, i), compute the response-level output-projection gradient

$$\nabla_{W_{\mathrm{out}}} \ell_{b,i} = \frac{1}{|o_{b,i}|} \sum_{t=1}^{|o_{b,i}|} \big(\nabla_{z_{b,i,t}} \ell_{b,i,t}\big) h_{b,i,t}^{\top},$$

and flatten it into a feature vector $g_{b,i}$.

4: **Batch Gram geometry.** Stack all features into

$$\mathbf{F} \leftarrow [g_{b,i}]_{(b,i)} \in \mathbb{R}^{d \times m}, \qquad K \leftarrow \frac{1}{m} \mathbf{F}^{\top} \mathbf{F} \in \mathbb{R}^{m \times m}.$$

5: **Dominant subspace.** Compute $K = V \Lambda V^{\top}$ and the regularized participation score

$$\mathrm{PR}(K) \leftarrow \frac{(\sum_{j=1}^{m} \lambda_j)^2}{\sum_{j=1}^{m} \lambda_j^2}.$$

Set

$$k_t \leftarrow \mathrm{clip}\big(\lfloor \mathrm{PR}(K) \rceil,\, 1,\, m\big), \qquad V_k \leftarrow [v_1, \ldots, v_{k_t}].$$

6: **Projectors.** Construct

$$P_k \leftarrow V_k V_k^{\top}, \qquad P_k^{\perp} \leftarrow I - P_k.$$

7: **Dual-channel advantage decomposition.**

$$a_{\mathrm{main}} \leftarrow P_k a, \qquad a_{\mathrm{exp}} \leftarrow P_k^{\perp} [a]_+.$$

8: **Signed-cancellation estimate.** Construct signed per-sample update contributions

$$u_{b,i} \propto \hat{A}_{b,i} \nabla_\theta \log \pi_\theta(o_{b,i}|q_b),$$

and compute

$$n_{\mathrm{eff}} \leftarrow \frac{\big\| \sum_{(b,i)} u_{b,i} \big\|_2^2}{\sum_{(b,i)} \|u_{b,i}\|_2^2 + \epsilon}.$$

9: **Adaptive mixing and policy update.** Set

$$\alpha_t \leftarrow \max(0, 1 - n_{\mathrm{eff}}), \qquad a' \leftarrow a_{\mathrm{main}} + \alpha_t a_{\mathrm{exp}} = P_k a + \alpha_t P_k^{\perp} [a]_+.$$

Update θ with the base GRPO-style objective, replacing each $\hat{A}_{b,i}$ by $a'_{b,i}$.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper clearly states its main claims and supports them with theory, diagnostics, and experiments.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation of the work is described in the conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Theoretical assumptions are stated, and the main residual-gradient bound is proved in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides datasets, models, training settings, hyperparameters, and the SALT algorithm.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The work uses public datasets and provides public code, training scripts, and reproduction instructions.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup, evaluation metrics, sampling settings, and implementation details are described.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Results are averaged over five runs and include paired confidence intervals.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports hardware, wall-clock time, and SALT overhead.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses public assets and automated evaluation without human-subject data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: This is a methodological RLVR optimization paper with no new datasets, human-subject experiments, deployment scenarios, or application-specific systems; any societal impact is indirect and tied to downstream model use.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No new high-risk dataset or deployed model system is released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: The draft does not yet explicitly list licenses for all used datasets, models, and tools.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The submission provides public code for SALT and the experimental pipeline.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work does not involve crowdsourcing or human-subject research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human-subject research is conducted.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are the main experimental subject and are described in the paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.