DISSECTING REINFORCEMENT LEARNING: MECHANISMS BEHIND COMPOSITIONAL REASONING IN LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) has become the standard way to train large reasoning models, yet it is unclear which of its components account for its advantage over supervised fine-tuning (SFT). We study this question through the lens of compositional generalization: how well the model learns to compose multiple atomic skills to solve harder tasks. Our main contribution involves organizing post-training methods along two axes. The *data* axis classifies methods by the source of training data (teacher, frozen checkpoint, or on-policy models). The *loss* axis, in the binary reward setting, considers how different loss functions place weight on correct and incorrect samples during gradient updates. Through this unifying perspective, we are able to isolate the effect of on-policy data, negative gradients, the group-mean baseline, and standard-deviation normalization individually. On a string-function task and on competition math, we find that neither on-policy data nor negative gradients suffices alone. However, when combined, they improve average accuracy and allow the model to generalize beyond the compositional depths or difficulty of math on which it was trained. We provide a theoretical intuition for why this is the case: methods that give weight to negative examples allow the gradient to concentrate at critical points which distinguish successes from failures, while on-policy data ensures that these updates concentrate on wrong decisions the model would actually make. Consistent with this account, we find that training dynamics and results correspond to several predicted failure cases. Positive-only training with off-policy data such as teacher SFT is stable but lacks generalization, while on-policy positive-only training can lead to entropy collapse and shortened responses which eventually degrade performance. Lastly, applying negative rewards to off-policy data destabilizes training with large gradient norms and produces degenerate outputs. Combining on-policy sampling with negative gradients avoids these observed failure modes and supports sustained generalization.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has become the standard recipe for training large reasoning models: the model samples candidate solutions, a verifier checks them, and the model is updated toward the correct ones without human labels (DeepSeek-AI, 2025; Luo et al., 2025; Zeng et al., 2025). Why this works better than supervised fine-tuning (SFT) on correct solutions is still debated. One view holds that RL lets the model *discover* new solutions through exploration (Wen et al., 2025), a second that it mostly *sharpens* probability mass the base model already has (Yue et al., 2025), and a third that it teaches the model to *chain* skills it already has (Setlur et al., 2025). Furthermore, some studies attribute RL's advantage to training on the model's own distribution (Chen et al., 2025; Ming et al., 2026), others to the negative gradient on incorrect samples (Tian et al., 2026; Setlur et al., 2025), and others to the group-relative advantage, which implicitly assigns step-level credit (Sullivan & Koller, 2026). It is difficult to separate these effects because an RL algorithm like GRPO differs greatly from SFT in that the data is on-policy, incorrect samples are penalized, and rewards are centered by a group baseline and scaled by the group standard deviation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We separate these factors with a controlled decomposition along two axes. The **data** axis is where training trajectories come from: a stronger teacher, the frozen initial checkpoint, or the current policy. The **loss** axis is how samples are weighted. In a simple binary reward setup, SFT, REINFORCE and GRPO are all weighted log-likelihood objectives that differ only in the weights they give to correct and incorrect samples. This lets us go from SFT to GRPO one component at a time: negative samples, group-mean baseline, and standard-deviation normalization. Crossing the two axes gives a 2×2 grid of methods with off-policy SFT and GRPO at opposite corners, and we examine the performance of representative post-training methods in this grid starting from the same baseline. We evaluate methods on *compositional generalization*, the ability to combine known atomic skills to solve harder problems, a setting where the gap between SFT and RL is large and easy to measure (Yuan et al., 2025). We first examine the string-function task of Yuan et al. (2025), where models are trained on compositions of two functions and evaluated on compositions of up to eight. We further examine competition math (Luo et al., 2025), where models are trained on easy and medium problems and evaluated on hard ones (Figure 1).

In summary, our main contributions are:

- **A controlled decomposition of post-training methods.** We organize methods by rollout source and outcome weighting, separating the effects of on-policy sampling, negative gradients, the group-mean baseline, and standard-deviation normalization.
- **Evidence that generalization gains extend beyond GRPO.** Across string composition and competition math, combining on-policy sampling with negative gradients supports sustained generalization, even with a simple REINFORCE objective using positive and negative rewards. The group-mean baseline provides substantial additional gains on string composition, while differences among negative-gradient losses are smaller on math.
- **A theoretical interpretation supported by training analyses.** Our analysis shows how group centering cancels shared gradient components and, under our assumptions, can focus updates on critical steps that separate success and failure. It also explains how rollout source changes the effect of reward weighting. Empirically, we identify distinct failure modes: limited extrapolation with off-policy SFT, entropy collapse and response shortening with on-policy SFT, and gradient instability with off-policy negative-gradient losses.

2 Compositional Generalization

2.1 Task Definition

Compositional generalization is the ability to compose multiple atomic skills to solve a harder problem. Generally, we can measure the difficulty of the task by the amount of composition required to solve it. To make this more concrete, we study a toy task where the goal is to predict the output of composed functions that transform strings (Yuan et al., 2025). We also extend our study to a more realistic task, competition math, where we test the model's ability to compose techniques learned in easier problems on a more difficult problem.

2.2 Toy Task: String Functions

Our first experiments are on the string-function setting as proposed in Yuan et al. (2025). The model receives definitions of simple atomic functions and must predict outputs of composed programs (Figure 1). Note that difficulty is controlled by composition depth. A level-i task requires composing i functions. The dataset contains 25 unique string functions, split into 13 in the train set and 12 in the eval set. With i composed functions, any combination of string functions is a unique task.

Following the training protocol proposed in Yuan et al. (2025), we *initialize* a pretrained model by training it on level-1 tasks with standard Supervised Fine-Tuning (SFT). This ensures that the model masters the atomic tasks. We call the resulting checkpoint **Base**, which is used as the starting point for compositional training. Our main training process involves training on level-2 tasks starting from **Base**. We use level-2 tasks since they are the lowest difficulty tasks that requires composition. We train using various post-training methods including SFT, DPO, and GRPO, and analyze their effectiveness. For evaluation, we measure performance on a separate evaluation set on tasks from level 1 to level 8. This allows us to see how performance extrapolates to higher composition levels.

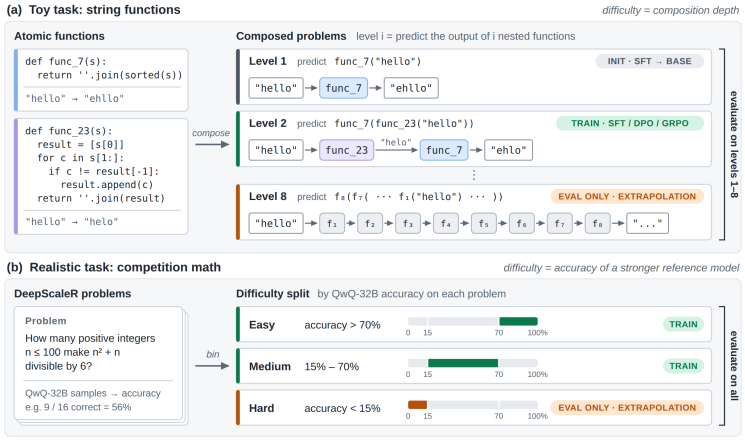


Figure 1: Overview of compositional tasks a) Toy task involving string transformation functions b) competition math task

2.3 Realistic Task: Math

We further study post-training algorithms with a more realistic task: mathematics. We use the training dataset from DeepScaleR (Luo et al., 2025), a collection of competition math problems. Following Setlur et al. (2025), we split the dataset into **Easy**, **Medium**, and **Hard** based on accuracy of a larger QwQ-32B model. The **Easy** set contains problems with accuracy greater than 70%, the **Hard** set contains problems with accuracy less than 15%, and the **Medium** set contains problems in between (Figure 1). We train on the easy and medium sets and evaluate on all three difficulties to see how the model generalizes to more difficulty problems.

3 Method: Decomposing Training Mechanisms

3.1 A Unified Two-Axis Framework

We propose a *two-axis unified framework* for organizing and analyzing the component-wise effects of various post-training algorithms. As a starting point, we identify the two core factors that differentiate SFT from GRPO: on-policy data and use of negative samples for gradient updates. Table 1 classifies known post-training methods into a two-axis grid, which allows us to place popular post-training algorithms like SFT, DPO, and GRPO. We generalize the two components into the following axes: a) **Data** axis involves the way we source and select data that will be used for training. b) **Loss function** axis involves the way we update the model parameters based on the data.

	Positive samples only	**Positive and negative samples**
Off-policy	SFT	DPO
On-policy	On-policy SFT	GRPO

Table 1: The unified two-axis framework with representative examples.

3.2 Data Axis

One common distinction between methods in RL is whether the data is on-policy or off-policy. On-policy data is generated by the current model being trained, while off-policy data is generated by a previous reference model, humans, or teacher models. To study the role of data source, we formalize the following settings: **(1) Teacher**: generated trajectories from a stronger teacher model. **(2) Bootstrap**: bootstrapped trajectories from the initial model checkpoint, and **(3) On-policy**: on-policy trajectories throughout training. We consider the **Teacher** and **Bootstrap** settings as Off-policy methods since they make gradient updates based on stale data.

3.3 Loss Function Axis

Our key insight is that we can view *loss functions as methods that apply different weights to positive and negative samples.* Table 2 summarizes how different loss functions weight positive and negative samples. Note that this understanding applies for tasks with binary reward ($r \in \{1, 0\}$).

Use of Negative Samples	Loss Function	Positive Weight	Negative Weight
No	SFT (1)	1	0
Yes	POS+NEG (2)	1	-1
	REINFORCE+ (3)	$1-\mu$	$-\mu$
	GRPO (4)	$(1-\mu)/\sigma$	$-\mu/\sigma$

Table 2: Comparison of loss functions by weights used for positive and negative samples.

First, we start with the standard loss function for Supervised Fine-Tuning (SFT).

$$\ell_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,y)}\left[\frac{1}{T}\sum_{t=1}^{T}\log \pi_\theta(y_t|x, y_{<t})\right]$$

For consistency with RL training procedures, we consider SFT as generating multiple rollouts for the same problem instance and filtering the positive samples for training. Let G denote the number of generated samples per problem. Furthermore, we can expand the log term to get:

$$\ell_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,\{y\}^{G^+})}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T}\log \pi_\theta(y_i|x)\right]$$

where G^+ denotes the set of positive samples.

Since SFT only uses positive samples, we can rewrite the loss as:

$$\ell_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,\{y\}^{G})}\left[\frac{1}{G}\sum_{i=1}^{G}\mathbf{1}[r_i = 1]\cdot\frac{1}{T}\log \pi_\theta(y_i|x)\right] \tag{1}$$

This derivation demonstrates that SFT is a special case where positive samples have weight 1 and negative samples have weight 0.

As a simple way to incorporate negative samples, we have the following loss function that increases likelihood of positive samples and decreases likelihood of negative samples:

$$\ell_{\text{POS+NEG}}(\theta) = -\mathbb{E}_{(x,\{y\}^{G})}\left[\frac{1}{G}\sum_{i=1}^{G}\mathbf{1}[r_i = 1]\cdot\log \pi_\theta(y_i|x) - \mathbf{1}[r_i = 0]\cdot\log \pi_\theta(y_i|x)\right] \tag{2}$$

This is essentially equivalent to REINFORCE (Williams, 1992) with rewards $r_i \in \{1, -1\}$ instead of $\{1, 0\}$. In other words, the loss function simply places $+1$ weight on positive samples and -1 weight on negative samples.

Next, we can further add a baseline to the REINFORCE loss where μ is the mean reward of the group:

$$\ell_{\text{REINFORCE+}}(\theta) = -\mathbb{E}_{(x,\{y\}^{G})}\left[\frac{1}{G}\sum_{i=1}^{G}(r_i - \mu)\cdot\log \pi_\theta(y_i|x)\right] \tag{3}$$

This takes into account the relationship of samples within G. If most samples in the group are correct, then the mean is high and the weight on each positive sample is low. On the other hand, if only a few samples are correct, the mean is low and the weight on positive samples is high. The same is true for negative samples. Hence, we can see loss (3) as placing $1-\mu$ weight on positive samples and $-\mu$ weight on negative samples.

Finally, we use standard deviation σ as a normalization factor to get to GRPO loss.

$$\ell_{\text{GRPO}}(\theta) = -\mathbb{E}_{(x,\{y\}^{G})}\left[\frac{1}{G}\sum_{i=1}^{G}\left(\frac{r_i - \mu}{\sigma}\right)\cdot\log \pi_\theta(y_i|x)\right] \tag{4}$$

For simplicity, we do not consider clipping and KL-divergence penalty. In this simplified version, we see that GRPO loss weighs the samples using a group-normalized advantage: $\frac{1-\mu}{\sigma}$ for positive samples and $\frac{-\mu}{\sigma}$ for negative samples.

4 THEORETICAL INTERPRETATION

In order to interpret the effects of different posttraining methods on generalization, we interpret rollouts as paths through a tree of possible reasoning steps, with nodes representing potential generated steps, and the root being the input prompt. Composition can be viewed as the successive correct traversal of multiple nodes, i.e. traversing the tree in a correct way. When successful and unsuccessful paths share upstream computations, contrasting their outcomes can suppress the shared successful prefix which is already learned, and retain signal at the step where the computation diverged. We note that this interpretation is related to that of Sullivan & Koller (2026), though our aim is to examine differences between GRPO and other posttraining methods.

We interpret the effects of reward weighting and rollout source through score-function updates, following the gradient view of Shao et al. (2024). Fix a prompt, a student checkpoint π_θ, and $K \geq 2$ independent rollouts from a behavior distribution ν, with binary rewards r_i. Let $s_{it} = \nabla_\theta \log \pi_\theta(y_{it} \mid x, y_{i,<t})$, and let $\bar{r}$ and $\bar{s}_t$ be group means. Full derivations are given in Appendix B.

Outcome contrast and centering. We first ask whether penalizing failures alone is enough to produce useful outcome contrasts, or if the group baseline is also necessary. We examine the simplest case distinguishing these hypotheses, POS+NEG.

For a group containing successes and failures, let Δ_t be the successful-minus-unsuccessful mean score at step t in the generation. Half-scaling the POS+NEG weights gives

$$\begin{aligned} \widehat{g}_t^{\mathrm{PN}} &= \frac{1}{K}\sum_i (r_i - \tfrac{1}{2}) s_{it} \\ &= \bar{r}(1-\bar{r})\Delta_t + (\bar{r} - \tfrac{1}{2})\bar{s}_t, \end{aligned} \tag{5}$$

while centering the update gives

$$\widehat{g}_t^{\mathrm{C}} = \frac{1}{K}\sum_i (r_i - \bar{r}) s_{it} = \bar{r}(1-\bar{r})\Delta_t. \tag{6}$$

In summary, we see that the difference between the two cases is that the centered update lacks the term $(\bar{r} - \frac{1}{2})\bar{s}_t$ which is a term that scales with the mean score and average reward of that group. However, both updates contain a term corresponding to the success-minus-failure contrast at that token position. Thus, both can emphasize decision points that distinguish successful and failed rollouts. When the group is balanced between successes and failures, both updates coincide.

The role of rollout source. The cancellation above holds regardless of rollout source, but its relevance to learning depends on which trajectories are compared. Our analysis shows that changing a fixed reward baseline preserves the expected update on-policy, but can change its direction off-policy. Thus, on-policy sampling contrasts successes and failures of the current model, whereas off-policy data may contain mistakes the model no longer makes and omits its current failures. This may intuitively lead to training difficulties as the mistakes emphasized may not be the ones the model actually tends to make. Full derivations appear in Appendix B.

5 EXPERIMENTS

5.1 SETUP

On the string function task, we use Llama-3.1-8B-Instruct as the backbone model, and first train it on level-1 tasks to ensure that the model has acquired the atomic skills. This setup stage uses Rejection Fine-Tuning (RFT) on 117k samples. We call the resulting checkpoint **Base**, which is used as the starting point for our main training methods. During compositional training, we train the **Base** model on level-2 tasks, and evaluate on levels up to 8. For the math task, we use Qwen3-1.7B as the base model, following the setup in Setlur et al. (2025). We train the model on first the **Easy** split followed by **Medium**. Throughout training, we evaluate performance on a held out validation set on all difficulties (**Easy** to **Hard**). For both tasks, we train the models with every combination of *data source* and *loss function* as defined in Section 3.3). Specific training hyperparameters are listed in Appendix A.

5.2 RESULTS

Table 3 reports accuracy by composition level on the string-function task, and Figure 2 summarizes the outcome of the full grid.

Data	Loss	L1	L2	L3	L4	L5	Avg (L1–L5)
–	Base (no training)	74.6	12.5	2.3	0.8	0.4	18.1
Bootstrap	SFT	74.2	24.2	5.9	2.7	1.2	21.6
Teacher	SFT	77.7	36.7	9.0	3.1	0.0	25.3
On-policy	SFT	69.1	28.5	6.6	2.7	1.2	21.6
On-policy	POS+NEG	78.1	54.3	13.3	4.7	0.8	30.2
On-policy	REINFORCE+Baseline	87.5	67.2	**27.7**	7.8	1.6	38.4
On-policy	GRPO	**88.3**	**73.0**	25.8	**11.7**	**2.3**	**40.2**

Table 3: String-function task: accuracy (%) by composition level. Off-policy runs with negative-sample losses collapse during training and are reported separately in Table 4.

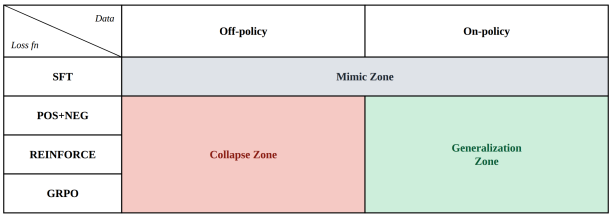


Figure 2: Outcome of every (data source, loss) combination. The grid separates into three regimes: SFT *mimics* its data, negative gradients on off-policy data *collapse*, and negative gradients on on-policy data *generalize*.

Averaged over levels 1–5, GRPO reaches 40.2%, more than doubling the Base model (18.1%) and well ahead of the best SFT variant (Teacher-SFT, 25.3%). The other on-policy losses with negative samples follow closely (REINFORCE+Baseline 38.4%, POS+NEG 30.2%), while every SFT variant shows modest improvement. The outcomes fall into three regimes (Figure 2).

(i) SFT mimics its data source, whether or not the data is on-policy. All three SFT variants improve level 2, the level they are trained on (12.5% $\rightarrow$ 24.2–36.7%), but the gains shrink quickly with depth. The quality of the data matters more than whether it is on-policy: Teacher-SFT, which imitates the long chain-of-thought trajectories of the GRPO checkpoint, beats Bootstrap and On-policy-SFT. On-policy data alone is therefore not what makes RL methods generalize.

(ii) Negative gradients on off-policy data collapse. When the same negative-sample losses are applied to off-policy data, training degenerates instead of improving (Table 4). For example, Bootstrap-GRPO falls from 12.2% at step 100 to 0.4%, with 99.4% of responses hitting the length cap without producing an answer. All runs end below the 12.5% level-2 accuracy that Base starts from. We analyze the mechanism of the collapse in Section 5.4.

Run	Accuracy	% length-capped
Bootstrap-GRPO	0.4%	99.4%
Bootstrap-REINFORCE+Baseline	9.6%	81%
Teacher-GRPO	8.8%	82%
Teacher-REINFORCE+Baseline	4.7%	93%

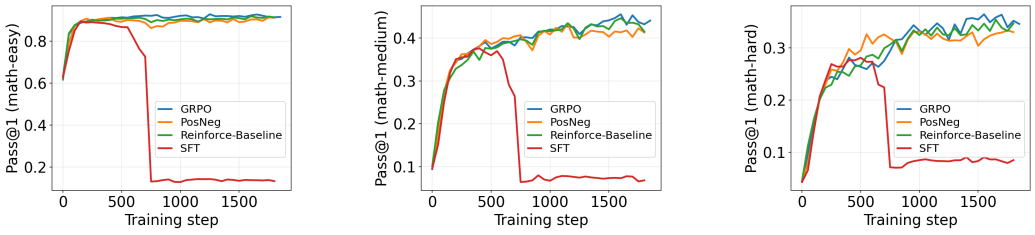
Table 4: Off-policy data with negative-sample losses: train-pool accuracy and the fraction of generations that hit the length cap without a terminated answer.

(iii) Negative gradients on on-policy data generalize. Only the combination of on-policy data and a loss that uses negative samples produces large gains beyond the training level, yielding accuracy in the 30.2–40.2% range.

Math. The same picture holds on the math task. An off-policy SFT method (e.g. Bootstrap-SFT) shows minimal improvement over the base model. Negative gradients with off-policy data (e.g. Bootstrap-GRPO) collapse to below the initial checkpoint. In the on-policy setting, negative-gradient losses generalize to the hard split, lifting it from 4.4% to 33.0–35.1% (Figure 3). On-policy SFT matches them early but collapses by the end of training.

Data	Loss	Easy	Medium	Hard
–	Base (no training)	62.6	9.7	4.4
Bootstrap	SFT	64.5	10.3	5.1
Bootstrap	GRPO	38.7	9.5	8.7
On-policy	SFT	86.8 $\rightarrow$ 14.3	36.0 $\rightarrow$ 7.8	28.1 $\rightarrow$ 8.3
On-policy	POS+NEG	91.3	41.3	33.0
On-policy	REINFORCE+Baseline	91.4	41.5	34.6
On-policy	GRPO	**91.7**	**44.1**	**35.1**

Table 5: Math task pass@1 (%) on easy, medium and hard validation splits. For On-policy SFT, we show best intermediate accuracy $\rightarrow$ final collapsed accuracy.



(a) Easy (b) Medium (c) Hard

Figure 3: Math task: validation pass@1 over training for each loss function in the on-policy setting.

5.3 Breaking down the components

On-policy SFT and GRPO see the same kind of data (rollouts from the current policy), yet they end 18.6% apart (21.6% vs. 40.2%). In order to understand the difference, we write all four on-policy losses in one form and note that the methods differ only in the weight w_i (Table 6). Each row adds exactly one component to the row above, so the accuracy differences between consecutive rows attribute the gap to individual components.

Loss	Weight w_i	Component added	Avg (L1–L5)	Δ
SFT	r_i	–	21.6%	–
POS+NEG	$2r_i - 1$	negative samples	30.2%	+8.6
REINFORCE+Baseline	$r_i - \bar{r}$	group-mean baseline	38.4%	+8.2
GRPO	$(r_i - \bar{r})/\sigma_r$	std. normalization	40.2%	+1.8

Table 6: The four on-policy losses as a ladder from SFT to GRPO on the string task. Accuracy is averaged over composition levels 1–5 on the string-function task.

Negative samples and the baseline each explain almost half of the gap; normalization explains little. Adding negative samples recovers 8.6% of the 18.6% gap (46%), adding the group-mean baseline recovers another 8.2% (44%), and standard-deviation normalization contributes the remaining 1.8% (10%). Naively pushing down incorrect samples is thus a large improvement over SFT but is far from sufficient, and the baseline, not the normalization, is what makes GRPO's use of negative samples effective. This empirically validates our theoretical interpretation from Section 4.

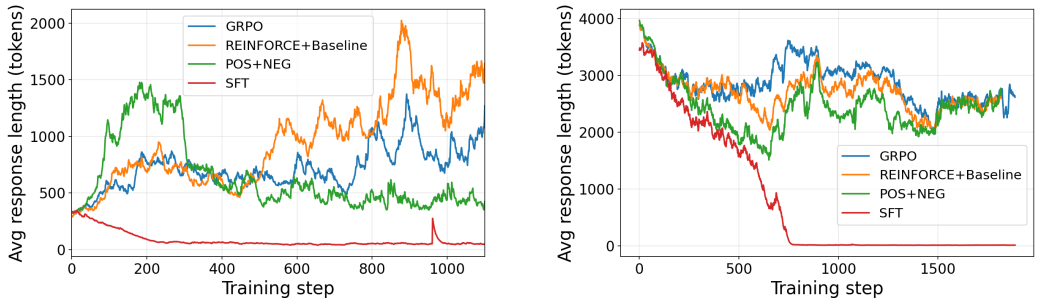
Math. On the math task the ordering is the same, with GRPO best overall, but the three negative-gradient losses are much closer to each other and the large step is from SFT to any loss with negative samples (Table 7).

Loss	Component added	Easy	Medium	Hard	Avg	Δ
SFT	–	14.3	7.8	8.3	10.1	–
POS+NEG	negative samples	91.3	41.3	33.0	55.2	+45.1
REINFORCE+Baseline	group-mean baseline	91.1	42.1	34.6	55.8	+0.6
GRPO	std. normalization	**91.7**	**44.1**	**35.1**	**57.0**	+1.2

Table 7: Math task, on-policy setting: pass@1 (%) on the easy, medium and hard validation splits at each run's final checkpoint, with the average over splits and the change from the row above.

5.4 Training Dynamics

We further analyze how each regime in Figure 2 arises during training, using response length, entropy and gradient norm.



(a) String-function task

(b) Math task

Figure 4: Average response length over training steps for each loss function in the on-policy setting.

Positive-only training contracts responses while negative samples sustain them. Figure 4 shows average response length during on-policy training. On-policy SFT shortens responses substantially on both tasks. With negative samples, response length is maintained on math and grows on the string-function task. In particular, for string functions, GRPO and REINFORCE+Baseline display steady growth throughout training while POS+NEG is less stable: response length rises rapidly early in training and then drifts back to the level of Base. This suggests that without a baseline, the balance between positive and negative updates shifts as the model's success rate changes during training, whereas the baseline keeps the two balanced at every step.

On-policy gradient norms are stable, except late in SFT. All four on-policy runs keep a gradient norm of 0.2–0.4 for most of training (Figure 8 in Appendix). Late in training, On-policy SFT shows increasingly frequent spikes of up to 5× the stable band, which the three negative-gradient losses do not. This coincides with the entropy collapse of the same run: once the output distribution is nearly deterministic, a single confident but wrong sample produces a disproportionately large gradient.

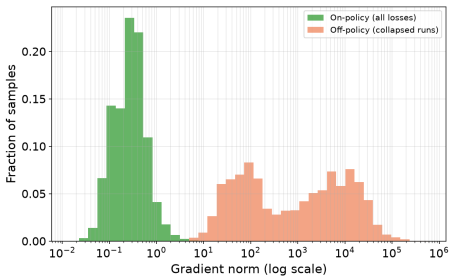


Figure 5: Distribution plot of the gradient norm (log scale).

Off-policy negative gradients blow up the gradient norm by four to five orders of magnitude. Figure 5 compares the gradient norms of the four collapsing runs from Table 4 with the on-policy runs. On-policy training stays within 0.1–0.5 throughout.

Every off-policy run leaves this range within the first 100–200 steps and reaches 10^3–10^5, peaking above 5×10^5 for Bootstrap-GRPO. A negative weight on a sample the current policy already assigns low probability has an unbounded objective, since the loss keeps decreasing as that probability goes to zero, and off-policy negatives become exactly such samples as the policy moves away from the data. The resulting models produce degenerate text such as token-level and sentence-level loops until the length cap (Appendix D).

In addition to the findings above, we found that SFT training trains the model to mimic the length distribution of the data source, and collapses entropy towards 0. More details on these results are in Appendix C.

6 Related Works

RL for reasoning: sharpening or discovery? A central question in LLM post-training has been whether RL primarily *sharpens* capabilities already present in the model by improving pass@1 while reducing the diversity of solutions (Yue et al., 2025; Wu et al., 2026; Zhao et al., 2025), or whether it can *discover* genuinely new reasoning strategies (Zeng et al., 2025). This distinction has motivated methods designed to preserve exploration during RL, including entropy regularization (Cui et al., 2025), representation-based novelty bonuses (Tuyls et al., 2026), and experience replay (Dou et al., 2025; Zhang et al., 2025b; Zhan et al., 2025).

Understanding the SFT–RL gap. Several works have sought to decompose the difference between SFT and RL. Although they can be viewed through a common policy-gradient framework, they differ in how data are generated, how outcomes are weighted, and how training is stabilized (Qin & Springenberg, 2025; Lv et al., 2026; Fu et al., 2025). Some works suggest that *on-policy data* matters (Ming et al., 2026) and GRPO without KL regularization or group normalization reduces to on-policy SFT (Zhao et al., 2026). Other works suggest that *negative gradients* play a big role as carefully selected negatives can improve out-of-domain generalization (Tian et al., 2026; Di et al., 2026) while Sullivan & Koller (2026) explains GRPO's group-relative advantages can act as an *implicit process reward*, reinforcing useful partial compositions that positive-only SFT cannot distinguish. Our paper provides a unified sample-weighting perspective that isolates the causal contribution of each component.

Compositional generalization. Multiple benchmarks show that LLMs struggle on *compositional generalization*: the ability to combine skills that a model can perform individually but has not learned to use together (Press et al., 2023; Zhao et al., 2024; Sun et al., 2025). Recent work suggests that RL can improve this ability by teaching models to compose learned string functions (Yuan et al., 2025). Other studies find that compositional curricula (Motwani et al., 2025), composable chain-of-thought data (Yin et al., 2025), and a more aligned mid-training (Zhang et al., 2025a) can help produce gains beyond simple capability sharpening.

7 Conclusion

In this work, we proposed a unified two-axis decomposition of post-training methods. The *data* axis ranges from teacher-distilled and bootstrapped off-policy data to fully on-policy rollouts, while the *loss* axis ranges from positive-only SFT through positive-plus-negative gradients, a group-mean baseline, and finally to the full GRPO objective. Our experiments found that neither on-policy data nor negative gradients suffice alone, and only when combined do they improve compositional ability. Contributions of components like negative gradients and baseline are task-dependent. Analysis of gradient norm and entropy gives insight into the training dynamics. These findings carry two broader implications. First, they support a *component-level* view of post-training rather than treating RL as a monolithic improvement over SFT. Second, they sharpen the ongoing debate over why RL works for reasoning. Our results are consistent with the view that negative gradients enable the model to suppress unproductive reasoning patterns rather than merely sharpen existing ones, and that on-policy data primarily serves to make this negative signal informative rather than to drive learning on its own.

AI USE STATEMENT

In this work, we used generative AI tools to design and provide feedback on research methodology or experiments, implement methods, and support qualitative and thematic data analysis. We have used GPT-6 Astra to check the proofs in this manuscript.

We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, or interpreting results. Furthermore, the tasks of generating synthetic data sets, assisting with translation, cleaning and reformatting datasets are not applicable to this work.

Additionally, we used generative AI tools for: create or modify scientific figures or images, create or edit software code, creation of artifacts, draft parts of a research paper, summarize or analyse existing literature, edit a research paper to improve readability, identify relevant literature, and propose a title or keywords for a research paper. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by the first author. LLM-generated artifacts were manually checked for accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We fully open-source the code and data used for this paper at `https://anonymous.4open.science/r/dissecting-rl-EB4E`. We further provide training details including hyperparamters and hardware used in Appendix A.

REFERENCES

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by Doing: The Role of On-Policy Data in Mitigating Forgetting, December 2025. URL http://arxiv.org/abs/2510.18874. arXiv:2510.18874 [cs].

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. SFT Memorizes, RL Generalizes: A Comparative Study of Foundation Model Post-training, May 2025. URL http://arxiv.org/abs/2501.17161. arXiv:2501.17161 [cs].

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The Entropy Mechanism of Reinforcement Learning for Reasoning Language Models, May 2025. URL http://arxiv.org/abs/2505.22617. arXiv:2505.22617 [cs].

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Zixiang Di, Jinyi Han, Shuo Zhang, Ying Liao, Zhi Li, Xiaofeng Ji, Yongqi Wang, Zheming Yang, Ming Gao, Bingdong Li, and Jie Wang. Not All Negative Samples Are Equal: LLMs Learn Better from Plausible Reasoning, February 2026. URL http://arxiv.org/abs/2602.03516. arXiv:2602.03516 [cs] version: 2.

Shihan Dou, Muling Wu, Jingwen Xu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. Improving RL Exploration for LLM Reasoning through Retrospective Replay, July 2025. URL http://arxiv.org/abs/2504.14363. arXiv:2504.14363 [cs].

Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. SRFT: A Single-Stage Method with Supervised and Reinforcement Fine-Tuning for Reasoning, June 2025. URL http://arxiv.org/abs/2506.19767. arXiv:2506.19767 [cs].

Song Lai, Haohan Zhao, Rong Feng, Changyi Ma, Wenzhuo Liu, Hongbo Zhao, Xi Lin, Dong Yi, Qingfu Zhang, Hongbin Liu, Gaofeng Meng, and Fei Zhu. Reinforcement Fine-Tuning Naturally

Mitigates Forgetting in Continual Post-Training, January 2026. URL http://arxiv.org/abs/2507.05386. arXiv:2507.05386 [cs].

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025. URL https://arxiv.org/abs/2503.20783.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005b 2025. Notion Blog.

Xingtai Lv, Yuxin Zuo, Youbang Sun, Hongyi Liu, Yuntian Wei, Zhekai Chen, Xuekai Zhu, Kaiyan Zhang, Bingning Wang, Ning Ding, and Bowen Zhou. Towards a Unified View of Large Language Model Post-Training, January 2026. URL http://arxiv.org/abs/2509.04419. arXiv:2509.04419 [cs].

Kohsei Matsutani, Shota Takashiro, Gouki Minegishi, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. RL Squeezes, SFT Expands: A Comparative Study of Reasoning LLMs, September 2025. URL http://arxiv.org/abs/2509.21128. arXiv:2509.21128 [cs].

Rui Ming, Haoyuan Wu, Shoubo Hu, Zhuolun He, and Bei Yu. One-Token Rollout: Guiding Supervised Fine-Tuning of LLMs with Policy Gradient, January 2026. URL http://arxiv.org/abs/2509.26313. arXiv:2509.26313 [cs].

Sumeet Ramesh Motwani, Alesia Ivanova, Ziyang Cai, Philip Torr, Riashat Islam, Shital Shah, Christian Schroeder de Witt, and Charles London. h1: Bootstrapping llms to reason over longer horizons via reinforcement learning, 2025. URL https://arxiv.org/abs/2510.07312.

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 E. Miller, Maddie Simens, Amanda Askell, P. Welinder, P. Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, March 2022. URL https://www.semanticscholar.org/paper/Training-language-models-to-follow-instructions-Ouyang-Wu/d766bffc357127e0dc86dd69561d5aeb520d6f4c.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. Measuring and Narrowing the Compositionality Gap in Language Models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.378. URL https://aclanthology.org/2023.findings-emnlp.378/.

Chongli Qin and Jost Tobias Springenberg. Supervised Fine Tuning on Curated Data is Reinforcement Learning (and can be improved), September 2025. URL http://arxiv.org/abs/2507.12856. arXiv:2507.12856 [cs].

Amrith Setlur, Matthew Y. R. Yang, Charlie Snell, Jeremy Greer, Ian Wu, Virginia Smith, Max Simchowitz, and Aviral Kumar. e3: Learning to Explore Enables Extrapolation of Test-Time Compute for LLMs, June 2025. URL http://arxiv.org/abs/2506.09026. arXiv:2506.09026 [cs].

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.

Michael Sullivan and Alexander Koller. GRPO is secretly a process reward model. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, 2026. URL https://arxiv.org/abs/2509.21154.

Yiyou Sun, Shawn Hu, Georgia Zhou, Ken Zheng, Hannaneh Hajishirzi, Nouha Dziri, and Dawn Song. OMEGA: Can LLMs Reason Outside the Box in Math? Evaluating Exploratory, Compositional, and Transformative Generalization, June 2025. URL http://arxiv.org/abs/2506.18880. arXiv:2506.18880 [cs].

Xueyun Tian, Minghua Ma, Bingbing Xu, Nuoyan Lyu, Wei Li, Heng Dong, Zheng Chu, Yuanzhuo Wang, and Huawei Shen. Learning from Mistakes: Negative Reasoning Samples Enhance Out-of-Domain Generalization, January 2026. URL http://arxiv.org/abs/2601.04992. arXiv:2601.04992 [cs] version: 1.

Jens Tuyls, Dylan J. Foster, Akshay Krishnamurthy, and Jordan T. Ash. Representation-based exploration for language models: From test-time to post-training, 2026. URL https://arxiv.org/abs/2510.11686.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, Jiang Bian, and Mao Yang. Reinforcement Learning with Verifiable Rewards Implicitly Incentivizes Correct Reasoning in Base LLMs, October 2025. URL http://arxiv.org/abs/2506.14245. arXiv:2506.14245 [cs].

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Fang Wu, Weihao Xuan, Ximing Lu, Mingjie Liu, Yi Dong, Zaid Harchaoui, and Yejin Choi. The Invisible Leash: Why RLVR May or May Not Escape Its Origin, January 2026. URL http://arxiv.org/abs/2507.14843. arXiv:2507.14843 [cs].

Yongliang Wu, Yizhou Zhou, Zhou Ziheng, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. On the Generalization of SFT: A Reinforcement Learning Perspective with Reward Rectification, October 2025. URL http://arxiv.org/abs/2508.05629. arXiv:2508.05629 [cs].

Fangcong Yin, Zeyu Leo Liu, Liu Leqi, Xi Ye, and Greg Durrett. Learning Composable Chains-of-Thought, May 2025. URL http://arxiv.org/abs/2505.22635. arXiv:2505.22635 [cs].

Lifan Yuan, Weize Chen, Yuchen Zhang, Ganqu Cui, Hanbin Wang, Ziming You, Ning Ding, Zhiyuan Liu, Maosong Sun, and Hao Peng. From $f(x)$ and $g(x)$ to $f(g(x))$: LLMs Learn New Skills in RL by Composing Old Ones, December 2025. URL http://arxiv.org/abs/2509.25123. arXiv:2509.25123 [cs].

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model?, May 2025. URL http://arxiv.org/abs/2504.13837. arXiv:2504.13837 [cs].

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping Reasoning With Reasoning, May 2022. URL http://arxiv.org/abs/2203.14465. arXiv:2203.14465 [cs].

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. SimpleRL-Zoo: Investigating and Taming Zero Reinforcement Learning for Open Base Models in the Wild, May 2025. URL http://arxiv.org/abs/2503.18892. arXiv:2503.18892 [cs].

Runzhe Zhan, Yafu Li, Zhi Wang, Xiaoye Qu, Dongrui Liu, Jing Shao, Derek F. Wong, and Yu Cheng. ExGRPO: Learning to Reason from Experience, October 2025. URL http://arxiv.org/abs/2510.02245. arXiv:2510.02245 [cs].

Charlie Zhang, Graham Neubig, and Xiang Yue. On the Interplay of Pre-Training, Mid-Training, and RL on Reasoning Language Models, December 2025a. URL http://arxiv.org/abs/2512.07783. arXiv:2512.07783 [cs].

Hongzhi Zhang, Jia Fu, Jingyuan Zhang, Kai Fu, Qi Wang, Fuzheng Zhang, and Guorui Zhou. RLEP: Reinforcement Learning with Experience Replay for LLM Reasoning, July 2025b. URL http://arxiv.org/abs/2507.07451. arXiv:2507.07451 [cs].

Anhao Zhao, Ziyang Chen, Junlong Tong, Yingqi Fan, Fanghua Ye, Shuhao Li, Yunpu Ma, Wenjie Li, and Xiaoyu Shen. On-Policy Supervised Fine-Tuning for Efficient Reasoning, February 2026. URL http://arxiv.org/abs/2602.13407. arXiv:2602.13407 [cs].

Jun Zhao, Jingqi Tong, Yurong Mou, Ming Zhang, Qi Zhang, and Xuanjing Huang. Exploring the Compositional Deficiency of Large Language Models in Mathematical Reasoning, October 2024. URL http://arxiv.org/abs/2405.06680. arXiv:2405.06680 [cs] version: 4.

Rosie Zhao, Alexandru Meterez, Sham Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. Echo Chamber: RL Post-training Amplifies Behaviors Learned in Pretraining, August 2025. URL http://arxiv.org/abs/2504.07912. arXiv:2504.07912 [cs].

A TRAINING DETAILS

All methods share the hyperparameters in Table 8; the two settings differ only in the values in Table 9. Each rollout batch is used for exactly one gradient step, with no KL penalty or entropy bonus. Teacher and bootstrap rollouts are generated once before training with the same sampling settings as on-policy rollouts.

Table 8: Hyperparameters shared across all runs.

Optimizer	AdamW, weight decay 0.01
Learning rate	1×10^{-6}, 5 warm-up steps, then constant
Gradient clipping	1.0
Prompts per step	16
Rollouts per prompt (G)	16
Sampling temperature	1.0
Max prompt / response length	1,024 / 4,096 tokens
Evaluation decoding	greedy
Hardware	4× A100 80GB, bf16, FSDP

Table 9: Setting-specific hyperparameters.

	String composition	**Competition math**
Initial model	Llama-3.1-8B-Instruct + depth-1 RFT	Qwen3-1.7B
Training steps	1,200	660 (easy) / 2,050 (easy+medium)
Evaluation frequency	every 100 steps	every 50 steps
Teacher checkpoint	on-policy GRPO	on-policy GRPO, step 1,800

String-task initialization SFT stage The string-task backbone model **Base** is obtained by fine-tuning Llama-3.1-8B-Instruct on correct sampled level-1 solutions with learning rate 2×10^{-5}, batch size 128, maximum length 3,072 tokens, for 2 epochs.

B DERIVATIONS AND SCOPE OF THE THEORETICAL INTERPRETATION

B.1 CONVENTIONS

All expectations condition on a fixed prompt and student checkpoint. The K rollouts are independent and identically distributed under a fixed behavior distribution ν. Rewards and sampled data are held fixed when differentiating the surrogate. For population identities, assume differentiability, parameter-independent support, interchange of expectation and differentiation, and finite required moments. A finite horizon with zero scores after termination handles unequal lengths. Scores are evaluated under the student even when another policy generated the data, with positive student probabilities on the behavior support. All norms are Euclidean in the chosen parameterization.

The main analysis assigns equal weight to trajectories and sums token scores. It addresses raw gradient contributions. Clipping, optimizer preconditioning, and regularization can change the actual parameter step. A semantic operation may be analyzed by summing its assigned token scores, but population score identities for a post-hoc segmentation require separate justification.

B.2 EXACT BATCH IDENTITIES

For a mixed-reward group, write μ_t^+ and μ_t^- for the mean scores among successes and failures, and set $\Delta_t = \mu_t^+ - \mu_t^-$. Since there are $K\bar{r}$ successes,

$$\begin{aligned} \widehat{g}_t^{\mathrm{C}} &= \bar{r}(1-\bar{r})\mu_t^+ - (1-\bar{r})\bar{r}\mu_t^- \\ &= \bar{r}(1-\bar{r})\Delta_t. \end{aligned} \tag{7}$$

Also,

$$\widehat{g}_t^{\mathrm{PN}} - \widehat{g}_t^{\mathrm{C}} = (\bar{r} - \tfrac{1}{2})\bar{s}_t, \tag{8}$$

which proves Eqs. equation 5–equation 6. Because $\sum_i (r_i - \bar{r}) = 0$,

$$\widehat{g}_t^{\mathrm{C}} = \frac{1}{K} \sum_i (r_i - \bar{r})(s_{it} - \bar{s}_t). \tag{9}$$

Thus any common vector component cancels. If all rewards agree, $\widehat{g}_t^{\mathrm{C}} = 0$ directly; the missing outcome-class mean and Δ_t are then undefined and unnecessary. These statements are deterministic and require no on-policy assumption.

B.3 Conditional concentration of contributions

Define the empirical reward standard deviation and score dispersion by

$$\sigma_r^2 = \bar{r}(1 - \bar{r}), \qquad D_t^2 = \frac{1}{K} \sum_i \|s_{it} - \bar{s}_t\|^2. \tag{10}$$

Vector Cauchy–Schwarz applied to Eq. equation 9 gives

$$\|\widehat{g}_t^{\mathrm{C}}\| \le \sigma_r D_t. \tag{11}$$

Consider upstream positions U and a compositional decision f in the same mixed-reward group. Suppose $D_u \le \varepsilon_u$ for $u \in U$ and $\|\Delta_f\| \ge \gamma > 0$. Let $E_U = (\sum_{u \in U} \varepsilon_u^2)^{1/2}$. Then

$$\frac{(\sum_{u \in U} \|\widehat{g}_u^{\mathrm{C}}\|^2)^{1/2}}{\|\widehat{g}_f^{\mathrm{C}}\|} \le \frac{E_U}{\sigma_r \gamma}. \tag{12}$$

The numerator is at most $\sigma_r E_U$ by Eq. equation 11, and the denominator is at least $\sigma_r^2 \gamma$ by Eq. equation 6.

For POS+NEG, set $e_t = (\bar{r} - \frac{1}{2})\bar{s}_t$ and $R_U = (\sum_{u \in U} \|e_u\|^2)^{1/2}$. The triangle and reverse-triangle inequalities give

$$\frac{(\sum_{u \in U} \|\widehat{g}_u^{\mathrm{PN}}\|^2)^{1/2}}{\|\widehat{g}_f^{\mathrm{PN}}\|} \le \frac{\sigma_r E_U + R_U}{\sigma_r^2 \gamma - \|e_f\|}, \tag{13}$$

provided $\sigma_r^2 \gamma > \|e_f\|$. This makes explicit why near balance alone is insufficient: the residual must be small in norm. Neither score similarity nor outcome separation follows from atomic accuracy alone. The bounds concern concatenated position contributions, not the norm of their sum or the behavioral effect of the update.

Indeed, for an upstream score s_u and a raw gradient step $\delta\theta = \eta \sum_t \widehat{g}_t$, first-order expansion gives

$$\delta \log \pi_\theta(y_u \mid h_u) = \eta s_u^\top \sum_t \widehat{g}_t + o(\eta). \tag{14}$$

This need not vanish when $\widehat{g}_u = 0$, because the parameters are shared across positions.

B.4 Expected updates and success filtering

By definition, $\mathbb{E}_\nu[r s_t] = C_{\nu,t} + a_\nu m_{\nu,t}$, proving the centered update. Independence across rollouts gives

$$\mathbb{E}_\nu[\bar{r}\bar{s}_t] = \frac{1}{K}\mathbb{E}_\nu[r s_t] + \frac{K-1}{K} a_\nu m_{\nu,t}, \tag{15}$$

$$\mathbb{E}_\nu[\widehat{g}_t^{\mathrm{C}}] = \frac{K-1}{K}\big(\mathbb{E}_\nu[r s_t] - a_\nu m_{\nu,t}\big). \tag{16}$$

This proves the centered update, including its finite-group factor. Multiplication by $K/(K-1)$ is equivalent to using each rollout's leave-one-out reward baseline. These expectations include homogeneous groups; selecting only mixed groups or dynamically filtering prompts requires accounting for the resulting change in the averaging distribution.

For genuinely on-policy samples, the conditional score identity is

$$\mathbb{E}_{y \sim \pi_\theta(\cdot|h)}[\nabla_\theta \log \pi_\theta(y \mid h)] = \nabla_\theta \sum_y \pi_\theta(y \mid h) = 0. \tag{17}$$

Averaging over prefixes gives $m_{\nu,t} = 0$, recovering standard policy-gradient baseline invariance Williams (1992). Thus positive-only, half-scaled POS+NEG, and bias-corrected centered estimators have equal means in this setting. Their finite-sample distributions and optimizer-induced dynamics can differ.

For $a_\nu > 0$, the expected score of an accepted success instead is

$$\mathbb{E}_\nu[s_t \mid r = 1] = \frac{C_{\nu,t}}{a_\nu} + m_{\nu,t}. \tag{18}$$

This describes an individual draw from the success-conditioned distribution, not every possible finite-batch filtering procedure. Normalizing separately by each prompt's accepted count can change relative prompt weights, so the fixed-prompt direction comparison does not automatically extend to an aggregate training objective. Off-policy, the sequence-summed mean score is $\sum_t m_{\nu,t} = \nabla_\theta \mathbb{E}_\nu[\log \pi_\theta(Y \mid x)]$ for fixed ν. It can contain useful imitation information; removing it is not automatically beneficial.

B.5 Relation to GRPO

A common group scaling gives $\widehat{g}_t^{\mathrm{N}} = \widehat{g}_t^{\mathrm{C}}/(\sigma_r + \tau)$, with $\tau \geq 0$ and the update defined as zero on homogeneous groups. It preserves within-group cancellation and relative contribution bounds. Because its denominator is random and group dependent, the unnormalized expectation identities do not directly apply to it. It also reweights prompts. Response-dependent weights w_i, such as $1/T_i$, retain a common score component c_t through

$$\frac{1}{K}\sum_i w_i(r_i - \bar{r})c_t = c_t \frac{1}{K}\sum_i w_i(r_i - \bar{r}), \tag{19}$$

which need not be zero. This is relevant to the normalization effects studied by Liu et al. (2025). Note that we do not analyze the full GRPO setup with clipping and other details, but our score surrogate here for simplicity.

C Additional Results for Training Dynamics Analysis

SFT reproduces the length distribution of its data. Figure 6 shows the distribution of response lengths on the evaluation set after SFT. The Bootstrap model stays close to Base, since its training data comes from the initial checkpoint, while the Teacher model adopts the long chain-of-thought trajectories of the GRPO checkpoint it was distilled from. This is the mechanism behind the Teacher-SFT advantage in Table 3: longer reasoning traces transfer better to unseen compositions, but the model only acquires them by imitation and does not improve beyond them.

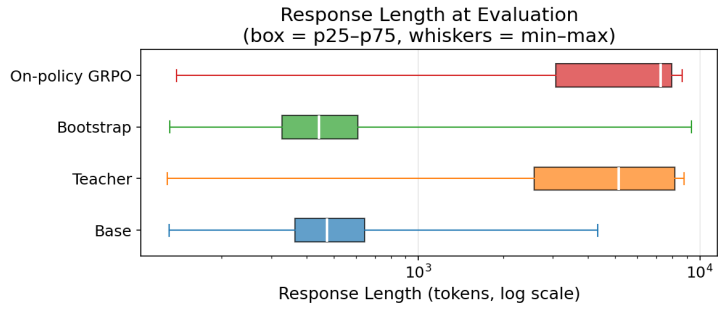


Figure 6: SFT on the string-function task: Response length distribution on the evaluation set.

SFT collapses entropy to zero; negative-gradient losses keep a non-zero floor. Figure 7 plots entropy on depth-2 validation rollouts. Under On-policy SFT, entropy reaches effectively zero by step ∼400 and stays there. POS+NEG, REINFORCE+Baseline and GRPO all hold a low but non-zero band (∼0.02–0.04) for the whole run. The contrast is between collapse to zero and a stable floor; entropy does not rise under any method, including GRPO.

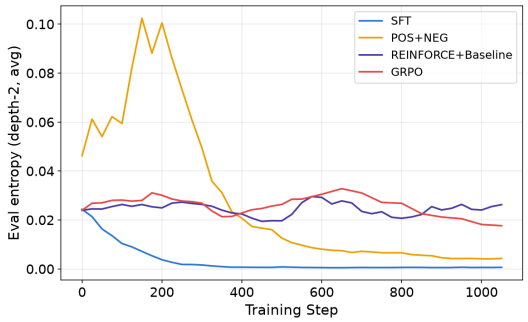


Figure 7: Entropy on the depth-2 validation set over training steps, on-policy setting.

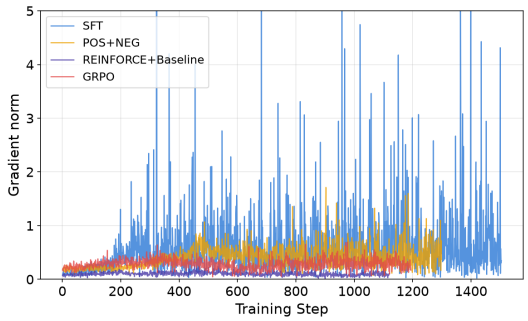


Figure 8: Gradient norm over training steps, on-policy setting (see Figure 5 for the off-policy comparison on a log scale).

D Examples of Degenerate Generations

Example Generation 1 (Bootstrap GRPO)

To predict the output of ‘main_solution(”kqqqqqq...(repeats q token until token limit)

Example Generation 2 (Teacher REINFORCE+Baseline)

If ‘s‘ is not a palindrome, ‘func_24‘ will return ‘s‘ if ‘s‘ is a palindrome after appending its reverse and recursively calling itself with ‘d - 1‘.

In this case, ‘s‘ is not a palindrome, but ‘func_24‘ will try to transform it into a palindrome.

(repeated 20x until token limit)

E Extended Related Works

RL for reasoning: sharpening or discovery? Reasoning post-training has evolved from bootstrapping on self-generated traces (Zelikman et al., 2022) and RLHF (Ouyang et al., 2022) to zero-RL directly from base models (DeepSeek-AI, 2025). A central question is whether RL primarily *sharpens* capabilities already present in the model by improving pass@1 while reducing the diversity of solutions (Yue et al., 2025; Wu et al., 2026; Zhao et al., 2025), or whether it can *discover* genuinely new reasoning strategies (Zeng et al., 2025). This distinction has motivated methods designed to preserve exploration during RL, including entropy regularization (Cui et al., 2025), representation-based novelty bonuses (Tuyls et al., 2026), and experience replay (Dou et al., 2025; Zhang et al., 2025b; Zhan et al., 2025).

Understanding the SFT–RL gap. A complementary line of work asks why RL can outperform SFT even when the two are trained on similar successful solutions. Empirically, SFT tends to memorize training distributions, whereas RL exhibits stronger out-of-distribution generalization (Chu et al., 2025). RL also appears to compress incorrect completions while SFT expands correct ones (Matsutani et al., 2025), and to retain more of the model's prior capabilities at matched performance (Chen et al., 2025; Lai et al., 2026; Wu et al., 2025). Several works have therefore sought to decompose the difference between SFT and RL. Although SFT, DPO, and RL can be viewed through a common policy-gradient framework, they differ in how data are generated, how outcomes are weighted, and how training is stabilized (Qin & Springenberg, 2025; Lv et al., 2026; Fu et al., 2025). Some works suggest that *on-policy data* matters (Ming et al., 2026) and GRPO without KL regularization or group normalization reduces to on-policy SFT (Zhao et al., 2026). Other works suggest that *negative gradients* play a big role as carefully selected negatives can improve out-of-domain generalization (Tian et al., 2026; Di et al., 2026) while Sullivan & Koller (2026) explains GRPO's group-relative advantages can act as an *implicit process reward*, reinforcing useful partial compositions that positive-only SFT cannot distinguish. Our paper provides a unified sample-weighting perspective that isolates the causal contribution of each component.

Compositional generalization. The distinction between sharpening and discovery is particularly important for *compositional generalization*: the ability to combine skills that a model can perform individually but has not learned to use together. Multiple benchmarks show that LLMs struggle to chain individually mastered skills (Press et al., 2023; Zhao et al., 2024; Sun et al., 2025). Recent work suggests that RL can improve this ability by teaching models to compose learned functions, such as $f(g(x))$ (Yuan et al., 2025). Other studies find that compositional curricula can produce gains beyond simple capability sharpening (Motwani et al., 2025), that explicitly composable chain-of-thought data facilitates skill composition (Yin et al., 2025), and that mid-training can provide the underlying competence on which subsequent RL builds (Zhang et al., 2025a).