Composition-RL: Compose Your Verifiable Prompts for Reinforcement Learning of Large Language Models

Xin Xu[1,2], **Clive Bai**[2], **Kai Yang**[2], **Tianhao Chen**[1], **Cai Yuchen**[2], **Yangkun Chen**[2], **Weijie Liu**[2], **Hao Chen**[6], **Yang Wang**[5], **Saiyong Yang**[2,†], **Can Yang**[1,3,4,7,†]

[1]Department of Mathematics, HKUST [2]Hy, Tencent [3]Big Data Bio-Intelligence Lab, HKUST
[4]State Key Laboratory of Nervous System Disorders, HKUST [5]Department of Mathematics, HKU
[6]Department of Computer Science and Engineering, HKUST [7]IAS Center for AI for Scientific Discoveries, HKUST

GitHub Repository 🤗 Hugging Face Models

Abstract

Large-scale verifiable prompts underpin the success of Reinforcement Learning with Verifiable Rewards (RLVR), but they contain many uninformative examples and are costly to expand further. Recent studies focus on better exploiting limited training data by prioritizing hard prompts whose rollout pass rate is 0. However, easy prompts with a pass rate of 1 also become increasingly prevalent as training progresses, thereby reducing the effective data size. To mitigate this, we propose *Composition-RL*, an offline data-augmentation approach that composes existing prompts before RL to address the pass-rate-1 bottleneck without modifying the RL optimizer. The composed dataset is generated and verified once from the fixed source pool and then used for RL training. Extensive experiments across model sizes from 4B to 30B show that *Composition-RL* consistently improves reasoning capability over RL trained on the original dataset. Performance can be further boosted with a curriculum variant of *Composition-RL* that gradually increases compositional depth over training. Additionally, *Composition-RL* enables more effective cross-domain RL by composing prompts drawn from different domains. All our code, data, and models are publicly available to facilitate future research.

1 Introduction

After the advent of OpenAI-o1 [1] and DeepSeek-R1 [2], Reinforcement Learning with Verifiable Rewards (RLVR) has reshaped the training lifecycle of large language models (LLMs), improving both text-only reasoning [3, 4, 5, 6] and multimodal question answering [7, 8]. Rapid progress in RLVR, including improved optimization algorithms [5, 9, 10, 11], more efficient training frameworks [12, 13, 14], and techniques to mitigate training–inference mismatch [15, 16], has contributed to the strong slow-thinking ability of large reasoning models (LRMs), often manifested as longer chain of thought (CoT) [17]. At its core, RLVR relies on large collections of training prompts paired with ground-truth answers to enable verifiable reward computation during training [3, 18, 19].

Prompts with $0/1$ rollout accuracy yield no gradient signals in RLVR algorithms [10], substantially reducing the number of available informative training prompts. However, collecting and cleaning additional high-quality prompts is expensive [19, 20]. To mitigate this, prior work primarily focused on hard prompts with zero success rate, via advantage shaping [9, 21], allocating more rollouts [22, 23], and hint-based augmentation [24, 25]. Nevertheless, while all-zero prompts constitute some fraction of the training set, as training progresses, an increasing proportion of prompts attain rollout accuracy of 1. This motivates the need for methods that can better leverage these 'easy' prompts.

[†]Corresponding authors.

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

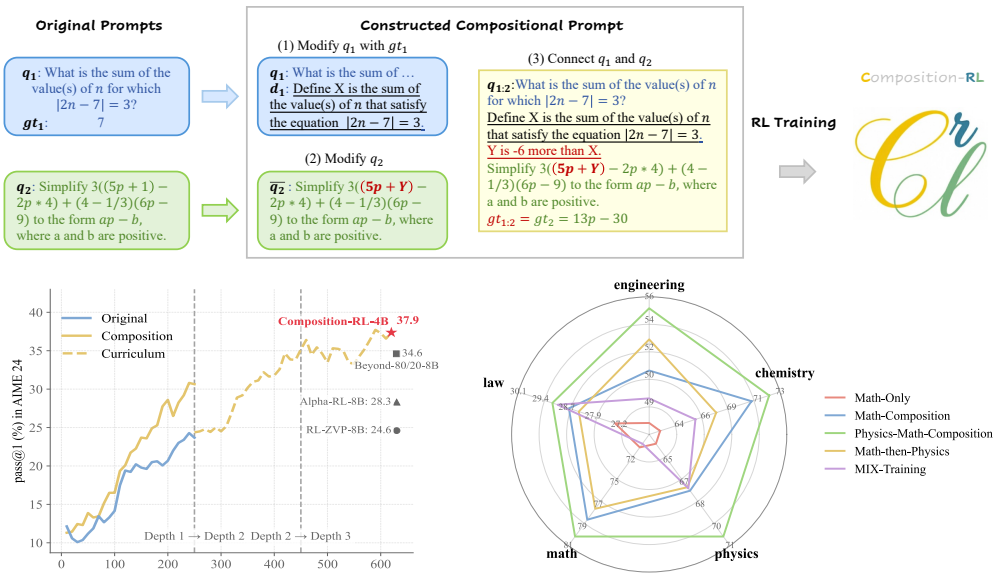

Figure 1: Overview of *Composition-RL*. **Top:** an example of composing two math problems, illustrating the high-level idea of *Composition-RL*. **Bottom left:** `pass@1` (%) on AIME24 versus training steps, summarizing key findings in Sections 5.2 and 5.3. **Bottom right:** cross-topic results on MMLU-Pro subjects with the top-5 largest sample sizes, highlighting the main finding in Section 5.4.

In this work, we propose *Composition-RL*, an offline data-augmentation framework that transforms a fixed pool of existing verifiable prompts into harder training data before RL. Following NCSP's sequential construction and LLM-based self-verification [26], we compose K source prompts and show that composition mitigates the growing number of "too-easy" prompts (Sections 4.1 and 4.2). We then train on the fixed compositional dataset with unmodified GRPO (Section 4.3); an overview is provided in Figure 1. As shown in Figure 1, *Composition-RL* outperforms RL training on the original prompts, with increasing performance gains when combined with a curriculum over compositional depth K. Moreover, composing prompts from different domains shows strong potential for cross-domain RL training. Our contributions are: ❶ We repurpose sequential prompt composition as offline RLVR data augmentation and introduce a pass-rate-1-motivated curriculum over compositional depth. ❷ Extensive experiments on 4B-30B LLMs demonstrate the effectiveness of *Composition-RL* and the curriculum variant of *Composition-RL*. ❸ We show that RL on composed prompts spanning physics and math is more effective than simply mixing training problems, regardless of whether sequential or joint training. ❹ We analyze the reasons behind the success of *Composition-RL* through the lenses of compositional generalization and implicit process supervision.

2 Preliminary

Notation. We denote an LLM parameterized by θ as a policy π_θ. Let q be an input query (i.e., a prompt) and $\mathcal{D}$ be the set of all queries. Given a response $r = (r_1, \ldots, r_{|r|})$ to q, the policy likelihood can be written as $\pi_\theta(r \mid q) = \prod_{t=1}^{|r|} \pi_\theta(r_t \mid q, r_{<t})$, where $r_{<t} = (r_1, \ldots, r_{t-1})$ and $|r|$ is the number of tokens in r. Each (q, r) can be evaluated by a verifier $v(q, r) \in \{0, 1\}$, which indicates whether r matches the ground-truth answer of q (denoted as gt).

RLVR. RLVR optimizes the expected verifiable reward: $\max_\theta \mathbb{E}_{q\sim\mathcal{D}}\big[\mathcal{J}_{\text{RLVR}}(\theta, q)\big]$ $(= \mathbb{E}_{q\sim\mathcal{D},\, r\sim\pi_\theta(\cdot|q)}\big[v(q, r)\big])$. A standard policy gradient estimator [27] is:

$$g_\theta(q, r) = A(q, r) \cdot \nabla_\theta \log \pi_\theta(r|q), \tag{1}$$

where $A(q, r) = v(q, r) - b(q)$ is called "advantage" and $b(q)$ is a baseline function that depends only on the query q. Group Relative Policy Optimization (GRPO) [28] approximates the advantage

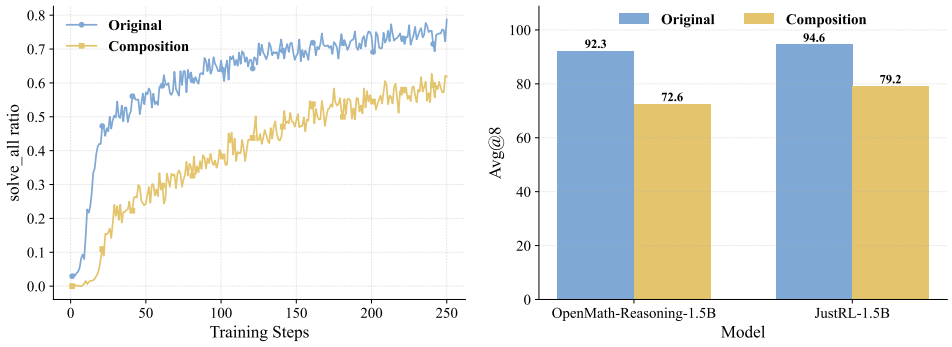

Figure 2: Visualization of meta-experiments. **Left:** *solve_all* ratio curve for RL of `Qwen3-4B-Base` with original prompts (MATH12K) versus compositional prompts. **Right:** `avg@8` accuracy on a subset of MATH500 and its corresponding compositional test prompts.

by sampling a group of G responses $\{r_1, \ldots, r_G\}$ from the old policy $\pi_{\theta_{\text{old}}}(\cdot \mid q)$:

$$\hat{A}_i = \frac{v(q, r_i) - \text{mean}\big(\{v(q, r_j)\}_{j=1}^{G}\big)}{\text{std}\big(\{v(q, r_j)\}_{j=1}^{G}\big)}. \tag{2}$$

Then the objective of GRPO becomes $\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q\sim\mathcal{D}}\big[\mathcal{J}_{\text{GRPO}}(\theta, q)\big]$ and $\mathcal{J}_{\text{GRPO}}(\theta, q)$ is:[1]:

$$\frac{1}{\sum_{i=1}^{G}|r_i|}\sum_{i=1}^{G}\sum_{t=1}^{|r_i|}\min\Big(i_{i,t}(\theta)\,\hat{A}_i,\ \text{clip}\big(i_{i,t}(\theta), 1-\epsilon, 1+\epsilon\big)\hat{A}_i\Big),\ i_{i,t}(\theta) = \frac{\pi_\theta(r_{i,t} \mid q, r_{i,<t})}{\pi_{\theta_{\text{old}}}(r_{i,t} \mid q, r_{i,<t})}. \tag{3}$$

Dynamic Sampling. In practice, GRPO objective can be approximated by $\hat{\mathcal{J}}_{\text{GRPO}}(\theta) = \mathbb{E}_{q\sim\mathcal{B}}\big[\mathcal{J}_{\text{GRPO}}(\theta, q)\big]$, where $\mathcal{B} \subset \mathcal{D}$ denotes a sampled mini-batch of prompts at a given training step. When a prompt has an empirical success rate of 0 or 1 (i.e., all sampled responses are incorrect or all are correct), its advantage is set to zero; consequently, by Equation (1), policy-gradient updates vanish. To mitigate this, dynamic sampling [10] first over-samples a larger candidate set $\hat{\mathcal{B}}$ and then constructs the training batch by filtering out uninformative prompts:

$$\mathcal{B} = \Big\{q \in \hat{\mathcal{B}} \ :\ 0 < \text{mean}\big(\{v(q, r_j)\}_{j=1}^{G}\big) < 1\Big\}. \tag{4}$$

Hereafter, we call a prompt *solve_all* if its sampled responses $\{r_j\}_{j=1}^{G}$ are all correct, and *solve_none* if they are all incorrect. Following [29, 21],we use "uninformative," "zero-variance," and "zero-advantage" prompts as synonyms for *solve_all* and *solve_none* prompts.

3 Related Work

Longer Training with Finite Prompts. Amid the surge of interest in RLVR [1, 2], many studies investigate how to enable longer and more stable training under a fixed prompt set [5, 30]. One line of work improves training stability from an algorithmic perspective [11, 31, 32]. Another line aims to exploit limited training data better, including filtering uninformative prompts [10, 29, 33], shaping advantages for zero-advantage prompts [9, 21, 34], and allocating more samples to harder prompts [22, 23]. Among these, hint-based problem augmentation [24, 25] is most closely related to our work: they use hints to transform originally hard prompts into easier ones. In contrast, we make easy prompts harder via compositional prompt generation.

Enlarging RLVR Training Prompts. The fuel of RLVR is its training prompts. A substantial body of work is devoted to collecting and curating high-quality data from diverse sources [18, 19, 35]. Synthesizing data from existing datasets has also been extensively studied, both for evaluation [36, 37] and for SFT [38, 39, 40]. NCSP [26] introduces the sequential construction and LLM-based self-verification used for mathematical evaluation. We follow that construction but use its pipeline to augment RL data, adding a pass-rate-1-motivated curriculum, cross-domain composition, and analyses

[1] We adopt a more-commonly used version with token-level normalization suggested by [10].

of the resulting gains. MathFusion [41] fuses multiple instructions for SFT, whereas MathForge [42] combines a difficulty-aware RL objective with single-question reformulation. In contrast, we retain the standard RL optimizer and compose multiple source prompts to reuse otherwise uninformative *solve_all* examples. More recently, several efforts have begun to synthesize prompts specifically for RL training [20, 43, 44, 45]. Unlike synthetic logic-only problems or game-like environments, our experiments focus on mathematical RL and cross-domain composition.

Compositional Generalization. Compositional generalization refers to a model's ability to recombine learned skills to solve novel tasks. It has been a longstanding focus in natural language processing [46, 47, 48]. Prior work often studies compositionality using controlled testbeds, such as Skill-Mix [49] for language tasks, compositional math benchmarks [50], or algorithmic tasks [51]. [52] show that composing textual skills can benefit SFT. [53] suggest that compositionality is important for RL to acquire new skills. However, their results are restricted to synthesized string-manipulation tasks. In comparison, we study compositional prompts as RL training data across model scales and analyze compositional generalization and implicit process supervision.

4 Methodology & Meta-Experiments

4.1 *SPC*: Sequential Prompt Composition

[53] studies the role of composition in RL using a synthetic string-transformation setting, and [26] evaluates LLM performance under the composition of two math problems. We extend this line of work by investigating how composing training prompts affects RL training. In this section, we describe Sequential Prompt Composition (*SPC*): we first define how to compose two prompts, and then generalize to composing K prompts. The whole composition process is illustrated in Figure 1.

Composing Two Prompts. Given two prompts q_1 and q_2 with ground-truth answers gt_1 and gt_2, we define a composition operator *Compose* that maps $(q_1, q_2; gt_1, gt_2)$ to a composed prompt $q_{1:2}$ with ground-truth answer $gt_{1:2}$:

$$q_{1:2},\ gt_{1:2} = \textit{Compose}(q_1,\ q_2;\ gt_1,\ gt_2). \tag{5}$$

This operator only requires q_1 to have an extractable numeric answer and q_2 to contain a replaceable numeric condition; neither source prompt must itself be a mathematics problem. The operator *Compose* consists of three steps (see Figure 1(a) for one concrete example):

❶ **Modify q_1 with gt_1.** Extract a numeric value from gt_1, denoted by v_1. We then introduce a natural-language definition d_1 that names this value in terms of (q_1, gt_1), and form $\bar{q}_1 = q_1 \oplus d_1$. For instance, if q_1 is "*What is the sum of the value(s) of n for which $|2n-7|=3$?*" and $gt_1 = 7$, we set $v_1 = 7$ and add a definition such as: "*Let X be the sum of the value(s) of n satisfying $|2n-7|=3$.*"

❷ **Modify q_2.** Extract a numeric value from q_2 and replace it with a new variable v_2, yielding $\bar{q}_2 = q_2(v_2)$. For example, if q_2 is "*Simplify $2((5p+1)-2p\cdot 4)+(4-1\div 3)(6p-9)$ to the form $ap-b$, where a and b are positive,*" we may choose the constant 1 as v_2 and replace it with variable name Y, obtaining "*Simplify $2((5p+Y)-2p\cdot 4)+(4-1\div 3)(6p-9)$ to the form $ap-b$...*"

❸ **Connect q_1 and q_2.** Compute $v_1 - v_2$ and express the relation between the two variables as a statement r. Continuing the example above, with $v_1 = 7$ and $v_2 = 1$, we have $v_1 - v_2 = 6$, so we can add a constraint r such as: "*Y is 6 less than X.*" The composed prompt is then $q_{1:2} = \bar{q}_1 \oplus r \oplus \bar{q}_2$. By construction, the answer of the composed prompt is $gt_{1:2} = gt_2$. This composition is asymmetric to the order of q_1 and q_2, and solving $q_{1:2}$ requires solving q_1 first and then q_2.

Composing K Prompts. More generally, we can compose K prompts into one prompt. Given $q_1, \ldots, q_K$ with ground-truth answers $gt_1, \ldots, gt_K$, Sequential Prompt Composition (*SPC*) applies *Compose* recursively for $K-1$ steps:

$$\begin{aligned} &\textit{SPC}(q_1, \ldots, q_K;\ gt_1, \ldots, gt_K) = \textit{Compose}(q_1,\ q_{2:K};\ gt_1,\ gt_{2:K}),\\ &\text{where}\quad (q_{2:K}, gt_{2:K}) = \textit{SPC}(q_2, \ldots, q_K;\ gt_2, \ldots, gt_K). \end{aligned}$$

Finally, we will get the composed prompt $q_{1:K}$ and its answer $gt_{1:K}$. We term K as the *Compositional Depth*. Intuitively, solving $q_{1:K}$ requires the model having the ability to solve all $\{q_k\}_{k=1}^{K}$. The process of composing 2 prompts can be viewed as a special case of *SPC* with $K=2$. Therefore, we do not distinguish between these two hereinafter.

4.2 Meta-Experiments & Observation

As collecting new high-quality, verifiable training prompts can be costly [19, 20], a growing body of work has focused on better leveraging uninformative prompts in RLVR [25, 23]. However, existing methods primarily target *solve_none* prompts. In this section, we conduct some meta-experiments and have the following key observations: ❶ Beyond *solve_none*, the increasing prevalence of *solve_all* prompts is another major impediment to effective RL training. ❷ *SPC* can make easy prompts harder and reduce the ratio of *solve_all* prompts.

Dilemma of Effective Training Prompts. As the policy model becomes stronger during RLVR, the proportion of *solve_all* prompts observed during rollouts increases. Figure 2 (Left) plots the *solve_all* rate across RL training steps for `Qwen3-4B-Base` on the MATH training set [54]. The *solve_all* ratio rises rapidly from near zero to over 50% within the first 50 steps and then stabilizes around 75%. Although dynamic sampling is enabled to remove zero-variance prompts, the actual effective size of the whole training set at later stages is reduced to roughly 3K prompts ($12{,}000 \times (1 - 0.75)$). In contrast, the *solve_none* ratio remains low (about 5%) at 250 steps. These results motivate methods that can deal with *solve_all* prompts, in addition to *solve_none* prompts.

***SPC* can nudge the last bits out of existing prompts.** Intuitively, *SPC* makes easy prompts harder. We empirically validate this on a subset of the MATH500 test set using `OpenMath-Reasoning-1.5B` [55] and `JustRL-1.5B` [30]; additional details are provided in Section A.2. As shown in Figure 2 (Right), switching to compositional prompts reduces $avg@8$ by 19.7% for `OpenMath-Reasoning-1.5B` and by 15.4% for `JustRL-1.5B`. The *solve_all* rate also drops substantially: from 81.5% to 41.4% for `OpenMath-Reasoning-1.5B`, and from 88.5% to 60.0% for `JustRL-1.5B`. These results suggest that *SPC* can effectively reduce *solve_all* prompts, potentially turn part of the original uninformative prompts useful again. Additionally, at $K = 2$, the ordered candidate space contains up to $|\mathcal{D}| \cdot (|\mathcal{D}| - 1)$ compositions and thus grows quadratically with the number of source prompts.

Note that `JustRL-1.5B` is obtained by RL training `OpenMath-Reasoning-1.5B`. Another interesting observation is that `JustRL-1.5B` improves performance both on the MATH500 subset (by 2.3%) and the compositional test set (by 6.6%). This suggests that RL training on normal prompts[2] can also improve performance on compositional prompts. This raises a natural question: *Does RL training on compositional prompts benefit performance on normal reasoning tasks?*

4.3 *Composition-RL*: RL with Compositional Data

Composition-RL is an offline data-augmentation framework rather than a new RL optimizer. The growing *solve_all* rate motivates composition and the depth curriculum, but does not define an online selection rule: every prompt in the fixed source pool remains eligible for composition. Given the original training set $\mathcal{D} = \{(q_i, gt_i)\}_{i=1}^{|\mathcal{D}|}$, we can construct a level-K compositional prompt set via the LLM-driven *SPC* procedure:

$$\mathcal{D}_{C_K} = \big\{(q, gt) \,:\, q, gt = \textit{SPC}(q_1, \ldots, q_K;\, gt_1, \ldots, gt_K),\ (q_k, gt_k) \in \mathcal{D},\ \ q_i \neq q_j\ \forall i \neq j\big\}.$$

Since the size of $\mathcal{D}_{C_K}$ can be extremely large, we instead use a smaller surrogate set:

$$\hat{\mathcal{D}}_{C_K} = \big\{(q, gt) \,:\, q, gt = \textit{SPC}(q_1, \ldots, q_K;\, gt_1, \ldots, gt_K),\ (q_k, gt_k) \in \mathcal{D}_k,\ \ q_i \neq q_j\ \forall i \neq j\big\}, \quad (6)$$

where each $\mathcal{D}_k$ is a small random subset of $\mathcal{D}$ and serve as the candidate set for q_k, i.e., $q_k \sim \mathcal{D}_k$. In practice, we set $|\mathcal{D}_k| = 20$ for $k = 1, \ldots, K-1$, and $\mathcal{D}_K = \mathcal{D}$. During this offline construction, each new $q_K \in \mathcal{D}$ is paired with independently resampled candidate subsets $\mathcal{D}_1, \ldots, \mathcal{D}_{K-1}$; after generation and verification, $\hat{\mathcal{D}}_{C_K}$ is fixed throughout RL training.

Composition-RL optimizes the objective: $\max_\theta \ \mathbb{E}_{q \sim \hat{\mathcal{D}}_{C_K}} \big[\mathcal{J}_{\text{RLVR}}(\theta, q)\big]$. We use the GRPO objective $\mathcal{J}_{\text{GRPO}}(\theta)$, advantage estimator, and importance ratio in Equations (2) and (3); only the prompt distribution changes to the fixed compositional dataset. Dynamic sampling during RL merely filters zero-variance prompts from training batches and does not generate new compositions online. Unless otherwise specified, we use $K = 2$ in all remaining experiments, and abbreviate $\mathcal{D}_{C_2}$ as $\mathcal{D}_C$.

[2] As compositional prompts differ substantially in structure from the original ones, we refer to prompts from all existing datasets as *normal* prompts for simplicity.

Table 1: Main Results. "Avg@k" denotes the average accuracy (%) over k random generations (i.e., pass@1). The rows "Depth 1 + 2" and "+ Depth 3" are the results of curriculum *Composition-RL* in Section 5.3. * indicates that the corresponding results are obtained using the same training set size as MATH12K. For *Composition-RL*, the starred row uses a 12K subset of MATH-Composition-199K, while the unstarred row uses the full set.

Method / Data	Mathematics (In-Domain)					Multi-Task (Out-Of-Domain)			Overall
	AIME 24 Avg@32	AIME 25 Avg@32	Beyond AIME Avg@8	IMOBench Avg@4	Overall Avg.	GPQA Avg@8	MMLU-Pro Avg@1	Overall Avg.	Overall Avg.
Qwen3-4B-Base									
GRPO	23.3	19.5	9.0	14.4	16.6	43.7	58.6	51.2	28.1
MetaMath*	12.7	9.7	3.9	8.1	8.6	43.9	58.5	51.2	22.8
SAND-Math*	25.6	20.5	8.9	15.4	17.6	45.4	57.9	51.7	29.0
MathFusion*	16.5	9.5	6.3	8.9	10.3	41.5	54.2	47.9	22.8
MathForge*	25.5	14.1	6.0	10.9	14.1	36.8	47.4	42.1	23.4
*Composition-RL**	27.3 ↑4.0	23.1 ↑3.6	9.0 ↑0.0	15.9 ↑1.5	18.8 ↑2.2	46.2 ↑2.5	60.1 ↑1.5	53.2 ↑2.0	30.3 ↑2.2
Composition-RL	30.5 ↑7.2	23.3 ↑3.8	12.6 ↑3.6	14.3 ↓0.1	20.2 ↑3.6	46.3 ↑2.6	61.4 ↑2.8	53.9 ↑2.7	31.4 ↑3.3
Depth 1 + 2	33.0 ↑9.7	27.8 ↑8.3	13.1 ↑4.1	20.1 ↑5.7	23.5 ↑6.9	48.3 ↑4.6	63.8 ↑5.2	56.1 ↑4.9	34.4 ↑6.3
+ Depth 3	37.9 ↑14.6	29.7 ↑10.2	14.6 ↑5.6	22.9 ↑8.5	26.3 ↑9.7	48.5 ↑4.8	64.5 ↑5.9	56.5 ↑5.3	36.4 ↑8.3
Qwen3-8B-Base									
GRPO	26.1	20.4	13.7	16.2	19.1	48.2	62.6	55.4	31.2
*Composition-RL**	29.2 ↑3.1	25.2 ↑4.8	13.3 ↓0.4	18.8 ↑2.6	21.6 ↑2.5	49.1 ↑0.9	65.9 ↑3.3	57.5 ↑2.1	33.6 ↑2.4
Composition-RL	36.9 ↑10.8	26.5 ↑6.1	13.9 ↑0.2	18.4 ↑2.2	23.9 ↑4.8	48.9 ↑0.7	64.5 ↑1.9	56.7 ↑1.3	34.9 ↑3.7
Qwen3-14B-Base									
GRPO	34.4	30.2	17.0	21.3	25.7	55.0	67.2	61.1	37.5
*Composition-RL**	43.3 ↑8.9	36.0 ↑5.8	18.6 ↑1.6	23.5 ↑2.2	30.4 ↑4.7	54.7 ↓0.3	69.3 ↑2.1	62.0 ↑0.9	40.9 ↑3.4
Composition-RL	44.5 ↑10.1	36.9 ↑6.7	19.7 ↑2.7	25.9 ↑4.6	31.8 ↑6.1	54.2 ↓0.8	69.3 ↑2.1	61.8 ↑0.7	41.8 ↑4.3
Qwen3-30B-A3B-Base									
GRPO	25.2	16.2	7.5	13.2	15.5	50.7	62.6	56.7	29.2
*Composition-RL**	47.7 ↑22.5	29.8 ↑13.6	20.1 ↑12.6	22.9 ↑9.7	30.1 ↑14.6	58.2 ↑7.5	65.6 ↑3.0	61.9 ↑5.2	40.7 ↑11.5
Composition-RL	46.4 ↑21.2	30.3 ↑14.1	19.5 ↑12.0	22.8 ↑9.6	29.8 ↑14.3	54.6 ↑3.9	64.6 ↑2.0	59.6 ↑2.9	39.7 ↑10.5

5 Experiments

5.1 Experimental Setting

In this section, we briefly summarize our experimental setup, including training procedures, baselines, and evaluation. Additional details are provided in Section B.

Training Details We conduct RL training using the VeRL codebase [12]. Unless otherwise specified, we use a unified set of hyperparameters (batch size 256, learning rate 1×10^{-6}, and no warm-up) and fixed rollout settings (temperature 1, top_p 1, top_k -1, 8 rollouts per problem, and a maximum output length of 16K tokens). We train Qwen3-4B-Base, Qwen3-8B-Base, Qwen3-14B-Base, and Qwen3-30B-A3B-Base on the MATH training set [54]. For the cross-topic experiments in Section 5.4, we use the physics subset of MegaScience [56]. For the verifier, we choose Math-Verify, a rule-based verifier. For a fair comparison, we enable dynamic sampling to filter uninformative prompts, ensuring that the effective batch size at each step remains constant across experiments.

Baselines. In Section 5.2 (see Table 1), we compare *Composition-RL* with standard RLVR on MATH12K under the same number of gradient updates. For *Composition-RL*, we construct approximately 199K compositional prompts, which we denote as MATH-Composition-199K. For the 4B setting, we also compare with data-augmentation baselines, including MetaMath [38, 57], SAND-Math [58], MathFusion [41], and MathForge [42]. For fairness, we additionally report results of *Composition-RL* under a controlled training set size (denoted by "*" in Table 1). For the cross-domain experiments in Section 5.4, we compare *Composition-RL* with two baselines: *Mix Training* (RL on a mixed dataset comprising MATH12K and the MegaScience Physics subset) and *Math-then-Physics* (continued RL on Physics starting from a MATH12K-trained checkpoint). Additional details are provided in Section B.2.

Evaluation Details. Our evaluation benchmarks include both in-domain (ID) math reasoning tasks, AIME24/25, BeyondAIME [59], and IMOBench [60], and out-of-domain (OOD) multi-task reasoning benchmarks, GPQA-Diamond [61] and MMLU-Pro [62]. Following [2], we sample multiple responses per problem (from 1 to 32, depending on the benchmark size) and report pass@1 accuracy. Following [4, 3, 63], we set the temperature to 0.6, top_p to 0.95, top_k to 20, and the maximum output length to 32K tokens. See also Section B.3 for details.

Table 2: Results of cross-topic experiments across multiple benchmarks. "Avg@k" denotes the average accuracy (%) over k random generations (i.e., pass@1). "MATH12K + Physics" corresponds to the *Mix Training* baseline, and "Physics after MATH12K" corresponds to the *Math-then-Physics* baseline. The final block uses 10K multidisciplinary prompts from SCP-116K [64].

Dataset	Mathematics					Multi-Task			Overall
	AIME 24 Avg@32	AIME 25 Avg@32	Beyond AIME Avg@8	IMOBench Avg@4	Overall Avg.	GPQA Avg@8	MMLU-Pro Avg@1	Overall Avg.	Overall Avg.
MATH12K	23.3	19.5	9.0	14.4	16.6	43.7	58.6	51.2	28.1
MATH12K + Physics	19.7	16.5	8.3	12.0	14.1	44.4	59.6	52.0	26.8
Physics after MATH12K	25.3	22.3	8.6	14.4	17.7	45.2	61.4	53.3	29.5
Physics-MATH-Composition-141K	**32.4**	**25.5**	**10.6**	**17.8**	**21.6**	**46.6**	**62.7**	**54.7**	**32.6**
SCP-116K Multidisciplinary Prompts (10K)									
Mix RL	10.10	9.79	4.25	8.31	8.11	44.87	60.58	52.73	22.98
Composition-RL	13.02	11.10	5.38	9.69	9.80	49.84	65.00	57.42	25.67

5.2 Compositional Prompts Are Beneficial to RLVR

To evaluate *Composition-RL*, we report main results in Table 1 with additional metrics provided in Section E. From Table 1, we have the following observations:

❶ RL on compositional prompts improves aggregate in-domain mathematics and OOD multi-task performance across all model sizes. Across all model sizes, *Composition-RL* improves the overall mathematics performance by $+3.6\%$, $+4.8\%$, $+6.1\%$, and $+14.3\%$ for Qwen3-4B/8B/14B/30B-A3B, respectively. With the full compositional set, gains reach $+21.2\%$ on AIME24, $+14.1\%$ on AIME25, $+12.0\%$ on Beyond AIME, and $+9.6\%$ on IMOBench. Moreover, *Composition-RL* also improves OOD performance, increasing the multi-task overall by $+2.7\%$, $+1.3\%$, $+0.7\%$, and $+2.9\%$, leading to overall average gains of $+3.3\%$, $+3.7\%$, $+4.3\%$, and $+10.5\%$ across the four base models. These substantial aggregate gains demonstrate the effectiveness of *Composition-RL*.

❷ The benefits of *Composition-RL* scale with model size, with larger models exhibiting substantially larger improvements, especially in mathematics. Overall gains increase from $+3.3\%$ (4B) and $+3.7\%$ (8B) to $+4.3\%$ (14B), and peak at $+10.5\%$ for Qwen3-30B-A3B. The scaling effect is most pronounced on in-domain mathematics: improvements rise from $+3.6\%$/$+4.8\%$/$+6.1\%$ to $+14.3\%$ as model size increases from 4B to 30B, whereas OOD multi-task gains are smaller but remain consistently positive. Notably, the MoE 30B-A3B model underperforms the 14B dense model, consistent with the fact that MoE activates only a subset of experts per token and can be more sensitive to routing and optimization under a fixed training budget; nevertheless, *Composition-RL* still yields large gains on this model. Overall, these results highlight the scaling potential of *Composition-RL*.

❸ *Composition-RL* still outperforms all baselines under the same training set size. Compared with RL training on the original dataset, *Composition-RL* with 12K compositional prompts yields gains ranging from 2.2% to 11.5% for models from 4B to 30B. For the 4B model, *Composition-RL* outperforms MetaMath, SAND-Math, MathFusion, and MathForge; the closest baseline, SAND-Math, is lower by 1.3% overall. This indicates that the gains are not explained solely by increasing the number of training prompts, even though the seed questions are drawn from the same MATH training set. Overall, *Composition-RL* achieves the best performance under a controlled dataset size, and its ability to generate substantially more training prompts from existing human-annotated data is another key advantage.

5.3 Curriculum RL to Higher Compositional Depth

We experiment with a curriculum version of *Composition-RL* that progressively increases the composition depth and continues RL training. As shown in Table 1 and Figure 1, we have:

❶ Curriculum *Composition-RL* can make full use of the original prompts, producing progressively stronger LRMs as the composition depth increases. Continuing RL with Depth 2 data after Depth 1 (i.e., the original MATH12K) yields substantial gains over the Depth 1 checkpoint, improving by $+9.7\%$ on AIME24 and $+5.2\%$ on MMLU-Pro. Moreover, the Depth 1$\rightarrow$Depth 2 curriculum even outperforms training directly on MATH-Composition-199K, delivering a further $+3.0\%$ improvement on average. Adding Depth 3 stage continues to improve both in-domain tasks and OOD questions, with a further $+2.0\%$ overall gain. Figure 1 presents the validation performance curves

Table 3: Results of the ablation study. $\mathcal{D}_1$ specifies the strategy for constructing the candidate set for q_1: "RAND" randomly samples 20 prompts, whereas "FULL" selects from the entire original prompt set $\mathcal{D}$. "Baseline" denotes RL training on the original prompts $\mathcal{D}$. "Direct Concat" denotes RL training on prompts that are directly concatenated from two prompts without composition. For results on the fully solvable subset, n is the number of rollouts. Additional blocks report the Feeding-v_1 control, standalone training at each composition depth, and adaptive Depth-2 composition.

$\mathcal{D}_1$	$\mathcal{D}_2$	Mathematics (In-Domain)					Multi-Task (Out-Of-Domain)			Overall
		AIME 24 Avg@32	AIME 25 Avg@32	Beyond AIME Avg@8	IMOBench Avg@4	Overall Avg.	GPQA Avg@8	MMLU-Pro Avg@1	Overall Avg.	Overall Avg.
Baseline		23.3	19.5	9.0	**14.4**	16.6	43.7	58.6	51.2	28.1
RAND	RAND	22.6	19.6	8.2	13.8	16.1	43.6	60.0	51.8	28.0
FULL	RAND	24.5	**23.4**	8.7	14.1	17.7	44.8	59.9	52.4	29.2
RAND	FULL	**30.5**	23.3	**12.6**	14.3	**20.2**	**46.3**	**61.4**	**53.9**	**31.4**
Direct Concat 12K		18.4	14.4	6.4	10.4	12.4	43.1	55.9	49.5	24.8
Composition-RL 12K		**27.3**	**23.1**	**9.0**	**15.9**	**18.8**	**46.2**	**60.1**	**53.2**	**30.3**
Feeding-v_1		22.8	19.8	9.2	13.9	16.4	44.0	58.1	51.1	28.0
Standalone Composition Depth and Adaptive Composition										
Depth 1		23.3	19.5	9.0	14.4	16.6	43.7	58.6	51.2	28.1
Depth 2		27.3	23.1	9.0	15.9	18.8	46.2	60.1	53.2	30.3
Depth 3		26.1	22.3	9.2	15.2	18.2	45.9	60.2	53.1	29.8
Adaptive Depth 2		28.2	22.3	9.2	16.3	19.0	46.7	59.9	53.3	30.4
Results on Fully Solvable Subset of MATH12K										
initial model		13.1	10.1	5.2	8.7	9.3	42	53.9	48.0	22.2
Baseline (n=32)		13.2	11.6	5.1	9.9	10.0	42.1	53.7	47.9	22.6
Composition-RL (n=8)		**17.1**	**15.6**	**6.1**	**10.5**	**12.3**	**42.5**	**57.6**	**50.1**	**24.9**

throughout the curriculum training process. In summary, these results imply that *Composition-RL* effectively converts limited prompts (with high *solve_all* rates) into more useful samples.

❷ **Standalone depth and adaptive composition.** The standalone controls in Table 3 improve from 28.1% overall at Depth 1 to 30.3% at Depth 2, while Depth 3 obtains 29.8%; thus, increasing depth alone does not yield monotonic gains. In the adaptive variant, when current-policy rollouts identify a source prompt as *solve_all*, we replace it with a pre-generated composition whose second constituent q_2 is that source prompt. Adaptive Depth 2 composition reaches 30.4%, essentially matching the standard Depth 2 result of 30.3%, so this simple adaptive construction provides no clear advantage over the offline construction.

❸ **Curriculum *Composition-RL* on a 4B model surpasses several 8B baselines, even under unfavorable settings.** As shown in Figure 1, our final `Composition-RL-4B` model achieves 37.9% on AIME24, outperforming Beyond-80/20-8B [65] (34.6%), Alpha-RL-8B [6] (28.3%), and RL-ZVP-8B [21] (24.6%). Notably, *Composition-RL* uses only MATH12K and `Qwen3-4B-Base`, whereas these baselines train on DAPO-MATH-17K and `Qwen3-8B-Base`. Additional details are provided in Section B.2. Even in this unfavorable setting, *Composition-RL* achieves stronger performance, underscoring the importance of fully leveraging existing training prompts via composition.

5.4 Potential For General Domains

Previously, *Composition-RL* considered only math problems,In this section, we explore whether it can compose problems across domains. Specifically, we sample q_1 from the physics subset and q_2 from MATH12K, yielding a cross-domain compositional dataset, `Physics-MATH-Composition-141K`. We compare against the *Mix Training* and *Math-then-Physics* baselines described in Section 5.1, with details in Section B.2. As shown in Table 2, we make the following observations:

❶ **Adding physics prompts for RL training improves multi-task reasoning performance.** Both the *Mix Training* and *Math-then-Physics* baselines improve GPQA and MMLU-Pro performance relative to training on MATH12K alone. On average, *Mix Training* increases the multi-task average by 0.8%, and *Math-then-Physics* yields a larger gain of 2.1%. Moreover, *Math-then-Physics* can further increase performance on math reasoning tasks, whereas *Mix Training* slightly degrades the math reasoning ability. These results suggest that, while incorporating physics data benefits multi-task performance, sequential training (math followed by physics) is more effective than mixed training across topics. As shown in Figure 1, adding physics prompts (via *Mix Training* or *Math-then-Physics*) consistently improves generalization to *law*, *engineering*, and *chemistry* compared to training solely on math (*Math-Only*). Interestingly, training on `MATH-Composition-199K` (*Math-Composition*) also yields generalization beyond the math domain.

❷ **Composing physics and math problems is more effective than naively combining physics and math prompts.** RL training on `Physics-MATH-Composition-141K` outperforms all baselines by a large margin. Specifically, our method achieves a $+1.3\%$ gain over *Math-then-Physics* and a $+4.1\%$ gain over training solely on MATH12K on MMLU-Pro. On AIME24, it improves by $+7.1\%$ over *Math-then-Physics* and by $+9.1\%$ over training solely on MATH12K. As shown in Figure 1, *Physics-Math-Composition* consistently delivers the best results on both in-domain subjects (math and physics) and OOD subjects (law, engineering, and chemistry). These results show that, in the evaluated math–physics setting, compositional training can improve both in-domain and OOD performance and motivate broader evaluation across domain pairs.

❸ **Multidisciplinary prompts.** We additionally construct 10K training prompts from SCP-116K [64] and compare composition with mixed RL. As shown in Table 2, *Composition-RL* improves over Mix RL on all six benchmarks and raises overall accuracy from 22.98% to 25.67%, providing evidence for broader domains of scientific reasoning.

6 Analysis

6.1 Ablation Study

About Candidate Sets $\mathcal{D}_k$. For $K = 2$, our default *Composition-RL* constructs each composed prompt by sampling q_1 from a small 20-prompt subset $\mathcal{D}_1$ that is resampled for each q_2, while drawing q_2 from the full prompt pool $\mathcal{D}_2 = \mathcal{D}$; we compare this with two alternatives: *A*) both $\mathcal{D}_1$ and $\mathcal{D}_2$ are small random subsets with $|\mathcal{D}_1| = |\mathcal{D}_2| = 500$, and *B*) $\mathcal{D}_1 = \mathcal{D}$ while $\mathcal{D}_2$ is a small 20-prompt subset resampled for each q_1, keeping the total amount of compositional data approximately matched. As shown in Table 3, our default setting achieves the best overall performance, outperforming *variant A* by $+3.4\%$ and *variant B* by $+2.2\%$, while *variant A* remains close to the baseline despite using only 1K seed prompts, suggesting the potential of *Composition-RL* in limited-data regimes. The advantage over *variant B* suggests that diversity in $\mathcal{D}_2$ is particularly important, likely because the composed prompt $q_{1:2}$ shares the final ground-truth answer with q_2 and RLVR verifies only the final answer, so using the full $\mathcal{D}_2 = \mathcal{D}$ provides broader answer-level training signals.

The Necessity of Composition. To validate the necessity of conditioning q_2 on gt_1, we conduct an ablation study in which q_1 and q_2 are directly concatenated using the same question pairs as *Composition-RL*. As reported in Table 3, this direct-concatenation variant performs even worse than RL training on the original dataset. Moreover, *Composition-RL* outperforms the direct-concatenation variant by 5.5% in overall accuracy. We believe this degradation stems from a distribution shift, as direct concatenation essentially combines two problems into a single prompt. In contrast, our *SPC* guarantees that the compositional prompt functions as "one prompt".

Intermediate-value control. The Feeding-v_1 control obtains 28.0% overall, below the 30.3% of *Composition-RL* under the same 12K setting (Table 3). Thus, the gain cannot be explained solely by exposing the model to the intermediate value; requiring the model to recover and use v_1 is important.

Effectiveness on Fully Solvable Prompts. We start from an intermediate checkpoint of the 4B model and retain only the solve-all prompts from MATH12K, resulting in 7.2K prompts. We then compare *Composition-RL* on this fully solvable subset against a DAPO-style baseline, in which additional trajectories are adaptively sampled until an incorrect answer is obtained, with a maximum of 32 rollouts. As shown in Table 3, this DAPO-style baseline improves the initial model by only a small margin ($+0.4\%$ overall). In comparison, *Composition-RL* yields a $+2.7\%$ improvement even on this fully solvable subset, even with 8 rollouts per prompt. These results show that *Composition-RL* remains effective when applied from an intermediate checkpoint, even when the retained prompts are relatively simple.

6.2 Why *Composition-RL* Works

In this section, we further investigate why *Composition-RL* works. We analyze it from 2 perspectives:

Compositional Generalization. Compositional data may incentivize the acquisition of new skills. [53] show that, in a controlled synthetic setting, training on compositional data can elicit new reasoning skills. Analogously, if we view an original problem as requiring a stack of skills, composing prompts can create training instances that demand skill recombination. As shown

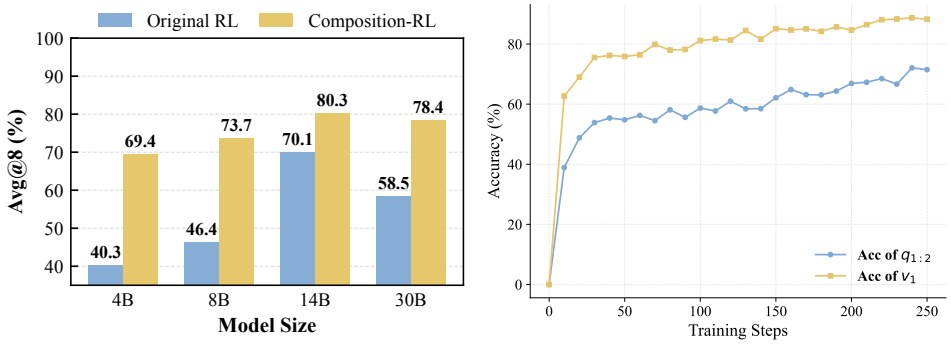

Figure 3: **Left**: `avg@8` accuracy on Depth-3 compositional test prompts across different model sizes. The bars compare Original RL with Composition-RL, i.e., whether Depth-2 compositional prompts are used during RL training. **Right**: The fraction of prompts for which $q_{1:2}$ is solved correctly, and the accuracy of recovering v_1 at each training step.

in Figure 3 (left), *Composition-RL* substantially improves performance on Depth-3 compositional test prompts: for the 4B, 8B, 14B, and 30B models, `avg@8` increases from $40.27/46.42/70.09/58.49\%$ without composition to $69.39/73.73/80.35/78.38\%$ with composition. This result is consistent with cross-depth compositional generalization; together with the standard-test gains in Table 1, it suggests that the acquired skills transfer beyond the training compositions.

Implicit Process Supervision. The final-outcome reward for composed prompts also provides implicit signals for the solution process. As shown in Figure 1, to solve the composed prompt $q_{1:2}$, LLMs must first obtain v_1 and then use v_1 to solve $\bar{q}_2$. We posit that this structured dependency nudges the model toward a correct intermediate step, at least "halfway" through the reasoning. As illustrated in Figure 3 (right), the steady improvement in recovering v_1 provides evidence that composed prompts can serve as implicit process supervision, even when training relies only on the verification of the final answers. Across the analyzed rollouts, only 0.43% reach the correct final answer while recovering an incorrect v_1, indicating that successful outcomes rarely bypass the intended intermediate dependency. For more analysis, please refer to Section D.

7 Conclusion & Discussion

In this paper, we study how to maximize the utility of existing prompts for RL training. Comprehensive experiments across various model sizes show that *Composition-RL* consistently outperforms RL on the original prompts. We also demonstrate the potential of composing prompts from different topics. Our analysis suggests that compositional prompts can provide implicit process supervision by encouraging correct intermediate steps. Our code, compositional datasets, and trained models are publicly available to support future RL research. Promising future directions include: ❶ Extending beyond MATH12K by composing more challenging math training set like Polaris-53K. ❷ Expanding composition to cover more domains. ❸ Adapting *Composition-RL* to on-policy distillation [66].

Acknowledgments

This work was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Reference Number: AoE/E-601/24-N). We would like to thank all reviewers for their helpful suggestions in improving this paper.

References

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

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

[3] 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, 2025. Notion Blog.

[4] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[5] Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025.

[6] Yuchen Cai, Ding Cao, Xin Xu, Zijun Yao, Yuqing Huang, Zhenyu Tan, Benyi Zhang, Guiquan Liu, and Junfeng Fang. On predictability of reinforcement learning dynamics for large language models. *arXiv preprint arXiv:2510.00553*, 2025.

[7] Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Tiancheng Han, Botian Shi, Wenhai Wang, Junjun He, et al. Mm-eureka: Exploring the frontiers of multimodal reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2503.07365*, 2025.

[8] Tong Xiao, Xin Xu, Zhenya Huang, Hongyu Gao, Quan Liu, Qi Liu, and Enhong Chen. Advancing multimodal reasoning capabilities of multimodal large language models via visual perception reward. *arXiv preprint arXiv:2506.07218*, 2025.

[9] Gongrui Nan, Siye Chen, Jing Huang, Mengyu Lu, Dexun Wang, Chunmei Xie, Weiqi Xiong, Xianzhou Zeng, Qixuan Zhou, Yadong Li, et al. Ngrpo: Negative-enhanced group relative policy optimization. *arXiv preprint arXiv:2509.18851*, 2025.

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

[11] Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025.

[12] 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.

[13] Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, Tongkai Yang, Binhang Yuan, and Yi Wu. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *ArXiv*, 2025.

[14] Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An llm post-training framework for rl scaling. https://github.com/THUDM/slime, 2025. GitHub repository. Corresponding author: Xin Lv.

[15] Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient rl framework secretly brings you off-policy rl training, August 2025.

[16] Penghui Qi, Zichen Liu, Xiangxin Zhou, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Defeating the training-inference mismatch via fp16. *arXiv preprint arXiv:2510.26788*, 2025.

[17] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[18] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

[19] Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025.

[20] Zhiyuan Zeng, Hamish Ivison, Yiping Wang, Lifan Yuan, Shuyue Stella Li, Zhuorui Ye, Siting Li, Jacqueline He, Runlong Zhou, Tong Chen, et al. Rlve: Scaling up reinforcement learning for language models with adaptive verifiable environments. *arXiv preprint arXiv:2511.07317*, 2025.

[21] Thanh-Long V Le, Myeongho Jeon, Kim Vu, Viet Lai, and Eunho Yang. No prompt left behind: Exploiting zero-variance prompts in llm reinforcement learning via entropy-guided advantage shaping. *arXiv preprint arXiv:2509.21880*, 2025.

[22] Zhicheng Yang, Zhijiang Guo, Yinya Huang, Yongxin Wang, Dongchun Xie, Yiwei Wang, Xiaodan Liang, and Jing Tang. Depth-breadth synergy in rlvr: Unlocking llm reasoning gains with adaptive exploration. *arXiv preprint arXiv:2508.13755*, 2025.

[23] Ziniu Li, Congliang Chen, Tianyun Yang, Tian Ding, Ruoyu Sun, Ge Zhang, Wenhao Huang, and Zhi-Quan Luo. Knapsack rl: Unlocking exploration of llms via optimizing budget allocation. *arXiv preprint arXiv:2509.25849*, 2025.

[24] Justin Chih-Yao Chen, Becky Xiangyu Peng, Prafulla Kumar Choubey, Kung-Hsiang Huang, Jiaxin Zhang, Mohit Bansal, and Chien-Sheng Wu. Nudging the boundaries of llm reasoning. *arXiv preprint arXiv:2509.25666*, 2025.

[25] Jiazheng Li, Hongzhou Lin, Hong Lu, Kaiyue Wen, Zaiwen Yang, Jiaxuan Gao, Yi Wu, and Jingzhao Zhang. Questa: Expanding reasoning capacity in llms via question augmentation. *arXiv preprint arXiv:2507.13266*, 2025.

[26] Xisheng Xiao and Hanlin Zhao. From a and b to a+ b: Can large language models solve compositional math problems? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13068–13089, 2025.

[27] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

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

[29] Yun Qu, Qi Wang, Yixiu Mao, Vincent Tao Hu, Björn Ommer, and Xiangyang Ji. Can prompt difficulty be online predicted for accelerating rl finetuning of reasoning models? *arXiv preprint arXiv:2507.04632*, 2025.

[30] Bingxiang He, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, et al. Justrl: Scaling a 1.5 b llm with a simple rl recipe. *arXiv preprint arXiv:2512.16649*, 2025.

[31] Kai Yang, Xin Xu, Yangkun Chen, Weijie Liu, Jiafei Lyu, Zichuan Lin, Deheng Ye, and Saiyong Yang. Entropic: Towards stable long-term training of llms via entropy stabilization with proportional-integral control. *arXiv preprint arXiv:2511.15248*, 2025.

[32] Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

[33] Haizhong Zheng, Yang Zhou, Brian R Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. *arXiv preprint arXiv:2506.02177*, 2025.

[34] Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

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

[36] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR, 2023.

[37] Xin Xu, Tong Xiao, Zitong Chao, Zhenya Huang, Can Yang, and Yang Wang. Can llms solve longer math word problems better? *ArXiv preprint*, abs/2405.14804, 2024.

[38] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *ArXiv preprint*, abs/2309.12284, 2023.

[39] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

[40] Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *ArXiv preprint*, abs/2407.13690, 2024.

[41] Qizhi Pei, Lijun Wu, Zhuoshi Pan, Yu Li, Honglin Lin, Chenlin Ming, Xin Gao, Conghui He, and Rui Yan. Mathfusion: Enhancing mathematical problem-solving of llm through instruction fusion. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7400–7420, 2025.

[42] Yanqi Dai, Yuxiang Ji, Xiao Zhang, Yong Wang, Xiangxiang Chu, and Zhiwu Lu. Harder is better: Boosting mathematical reasoning via difficulty-aware grpo and multi-aspect question reformulation. *arXiv preprint arXiv:2601.20614*, 2026.

[43] Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.

[44] Peiji Li, Jiasheng Ye, Yongkang Chen, Yichuan Ma, Zijie Yu, Kedi Chen, Ganqu Cui, Haozhan Li, Jiacheng Chen, Chengqi Lyu, et al. Internbootcamp technical report: Boosting llm reasoning with verifiable task scaling. *arXiv preprint arXiv:2508.08636*, 2025.

[45] Zafir Stojanovski, Oliver Stanley, Joe Sharratt, Richard Jones, Abdulhakeem Adefioye, Jean Kaddour, and Andreas Köpf. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2505.24760*, 2025.

[46] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, et al. Measuring compositional generalization: A comprehensive method on realistic data. *arXiv preprint arXiv:1912.09713*, 2019.

[47] Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

[48] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pages 2873–2882. PMLR, 2018.

[49] Dingli Yu, Simran Kaur, Arushi Gupta, Jonah Brown-Cohen, Anirudh Goyal, and Sanjeev Arora. Skill-mix: A flexible and expandable family of evaluations for ai models. *arXiv preprint arXiv:2310.17567*, 2023.

[50] 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. *arXiv preprint arXiv:2506.18880*, 2025.

[51] Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36:70293–70332, 2023.

[52] Haoyu Zhao, Simran Kaur, Dingli Yu, Anirudh Goyal, and Sanjeev Arora. Can models learn skill composition from examples? *Advances in Neural Information Processing Systems*, 37:102393–102427, 2024.

[53] 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. *arXiv preprint arXiv:2509.25123*, 2025.

[54] 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. *arXiv preprint arXiv:2103.03874*, 2021.

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

[56] Run-Ze Fan, Zengzhi Wang, and Pengfei Liu. Megascience: Pushing the frontiers of post-training datasets for science reasoning. *arXiv preprint arXiv:2507.16812*, 2025.

[57] Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2732–2747, 2024.

[58] Chaitanya Manem, Pratik Prabhanjan Brahma, Prakamya Mishra, Zicheng Liu, and Emad Barsoum. Sand-math: Using llms to generate novel, difficult and useful mathematics questions and answers. *arXiv preprint arXiv:2507.20527*, 2025.

[59] ByteDance-Seed. Beyondaime: Advancing math reasoning evaluation beyond high school olympiads. https://huggingface.co/datasets/ByteDance-Seed/BeyondAIME, 2025. Hugging Face repository.

[60] Minh-Thang Luong, Dawsen Hwang, Hoang H Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, et al. Towards robust mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 35406–35430, 2025.

[61] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[62] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

[63] Xin Xu, Cliveb AI, Kai Yang, Tianhao Chen, Yang Wang, Saiyong Yang, and Can Yang. Thinking-free policy initialization makes distilled reasoning models more effective and efficient reasoners. *arXiv preprint arXiv:2509.26226*, 2025.

[64] Dakuan Lu, Xiaoyu Tan, Rui Xu, Tianchu Yao, Chao Qu, Wei Chu, Yinghui Xu, and Yuan Qi. Scp-116k: A high-quality problem-solution dataset and a generalized pipeline for automated extraction in the higher education science domain. *arXiv preprint arXiv:2501.15587*, 2025.

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

[66] Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. https://thinkingmachines.ai/blog/on-policy-distillation.

[67] Qwen Team. Qwen2.5: A party of foundation models, September 2024.

[68] Xin Xu, Qiyun Xu, Tong Xiao, Tianhao Chen, Yuchen Yan, Jiaxin Zhang, Shizhe Diao, Can Yang, and Yang Wang. Ugphysics: A comprehensive benchmark for undergraduate physics reasoning with large language models. *arXiv preprint arXiv:2502.00334*, 2025.

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

[70] Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. Octothinker: Mid-training incentivizes reinforcement learning scaling. *arXiv preprint arXiv:2506.20512*, 2025.

Limitation

Most experiments use Depth 2, and evidence at higher depths is currently limited to Depth 3. The datasets are constructed offline from a fixed prompt pool; a simple adaptive variant performs on par with the standard construction, so stronger adaptive composition remains future work. Our construction and LLM-based self-verification follow NCSP [26] and therefore inherit its reliance on automated judgments; without a systematic human audit, some invalid prompts may remain after filtering. The method requires q_1 to yield an extractable numeric answer and q_2 to contain a replaceable numeric condition, although neither source prompt must itself be mathematical; extension to purely textual reasoning and unverifiable domains remains open. Finally, released data or models could be misused; we document dataset construction and intended use to mitigate this risk.

Impact Statement

This paper presents *Composition-RL*, which aims to advance research on RLVR. Our two compositional datasets, `MATH-Composition-199K` and `Physics-MATH-Composition-141K`, together with our code and trained models, are publicly available resources for future work on RL for LLMs. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Declaration of LLM Usage

In this paper, we use `Qwen2.5-32B-Instruct` [67] to generate our training data for *Composition-RL* and `Qwen3-Base` [4] models as the initial models for RL training. We also use `GPT-5.4` to polish the wordings and check for syntax.

A Details of Meta-Experiments

A.1 The *solve_all* Ratio

The training setup in Figure 2 (Left) follows the main experimental protocol (see Section B.1). We define the *solve_all* ratio as the fraction of *solve_all* prompts among all over-sampled prompts collected during dynamic-sampling rollouts at a given training step. For compositional data construction, please refer to Section F. For RL training details with compositional data, please refer to Section B.1 and Section B.2.

A.2 Details of Initial Evaluation For *SPC*

For the evaluation in Figure 2 (Right), we randomly sample 200 questions from MATH500 as seed prompts and use *SPC* to construct level-2 compositional prompts. Specifically, we form pairs (q_1, q_2) by sampling 5 seed questions as candidates for q_1 and pairing each with the 200 seed questions as q_2. After filtering, this procedure yields approximately 400 compositional test prompts. We use the same decoding settings as in Section B.3.

B Experimental Details

B.1 Training Details

In Section 5.1, we briefly describe the training setup. This appendix provides additional details on our RL training configuration. Two settings are particularly important: we enable dynamic sampling [10] to filter uninformative prompts, and we use rollout correction to mitigate training–inference mismatch [15].

Hyperparameters and Rollout Settings. Unless otherwise specified, we use a unified set of hyperparameters: batch size 256 (rollout batch size is identical to the mini-batch size), learning rate 1×10^{-6}, and no warm-up. We also adopt a unified rollout configuration: temperature 1, `top_p` 1, `top_k` -1, 8 rollouts per problem, and a maximum output length of 16K tokens.

Datasets and Verifiers. For the main experiments in Section 5.2 and Section 5.3, we train on the MATH training set [54]. Following the standard protocol, we exclude the MATH500 test set, leaving roughly 12K training prompts spanning five difficulty levels. For the cross-topic experiments in Section 5.4, we utilize the physics subset of MegaScience [56], which comprises approximately 23K prompts.

For training efficiency, we use Math-Verify as the verifier. Considering that rule-based verifiers do not reliably evaluate model outputs on physics problems [68], we filter the MegaScience physics subset by removing examples for which all eight responses from `Qwen3-4B-Thinking-2507` are judged incorrect by Math-Verify. This yields approximately 8.2K prompts on which rule-based verification is reliable.

B.2 Baselines

In this appendix, we provide additional baseline details for the experiments in Sections 5.2 to 5.4.

For the experiments in Section 5.2, the baseline corresponds to RL training on the original MATH12K training set, which contains 12K training prompts. Our *Composition-RL* trains instead on compositional prompts constructed from MATH12K. As described in Section 4.3, for each q_2 drawn from the full dataset, we independently resample a uniform 20-prompt subset from which q_1 is drawn, yielding $20 \times 12\text{K} = 240\text{K}$ compositional prompts in principle. As discussed in Section F.1, we apply a verification-and-filtering procedure to improve data quality. After verification of Step 1, approximately 231K prompts remain; after verifying Step 2, this is reduced to roughly 200K; and after the final check, we obtain about 199K compositional prompts. We refer to this composition set as `MATH-Composition-199K`.

For the experiments in Section 5.3, we include Beyond-80/20 [65], AlphaRL [6], and RL-ZVP [21] as additional reference baselines. We use the models that are initialized from `Qwen3-8B-Base` and trained on DAPO-MATH-17K prompts from these works. We report their results as RL-zero baselines to our curriculum-based *Composition-RL* trained from `Qwen3-4B-Base`. This comparison is unfavorable to *Composition-RL* due to differences in both model scale and training data. For curriculum *Composition-RL*, we first train `Qwen3-4B-Base` on the original MATH12K set (Depth 1). After performance saturates, we switch to the Depth 2 training set (`MATH-Composition-199K`), and then to the Depth 3 training set once Depth 2 saturates. The construction of the Depth 3 compositional set follows the procedure used for Depth 2. Since Beyond-80/20 [65], AlphaRL [6], and RL-ZVP [21] have not released their models at the time we were writing our paper, we report their results as quoted directly from the corresponding papers.

For the experiments in Section 5.4, we consider two natural RL baselines. The first is RL training on a mixture of MATH12K and the MegaScience Physics subset, which we denote as *Mix Training*. The second baseline continues RL training on Physics data, starting from a checkpoint trained on MATH12K, which we denote as *Math-then-Physics*. For *Math-then-Physics*, we train until performance saturates. For a fair comparison, we train *Mix Training* for approximately the same number of total gradient updates as the combined MATH12K stage plus the physics training stage. For *Composition-RL*, we sample q_1 from Physics and q_2 from MATH12K. After filtering, the resulting compositional dataset contains approximately 141K prompts, which we denote as `Physics-MATH-Composition-141K`.

B.3 Evaluation Details

To comprehensively evaluate model capabilities, we use a diverse suite of benchmarks spanning mathematical reasoning and multi-task reasoning:

1. **Mathematical reasoning:** We evaluate on AIME24, AIME25, BeyondAIME [59], and IMO-Bench [60]. Since AIME24 and AIME25 each contain 30 problems, we report `pass@1` using 32 samples per problem (`avg@32`). BeyondAIME contains 100 problems; we report `avg@8`. For IMO-Bench, we use the AnswerBench subset to enable rule-based verification; it contains 400 problems, and we report `avg@4`.
2. **Multi-task reasoning:** We evaluate on GPQA-Diamond [61] (approximately 200 problems) and report `pass@1` using 8 samples per problem. We also evaluate on MMLU-Pro [62]; since it contains over 5K problems, we report results from a single run.

Table 4: Difficulty-stratified evaluation (%). "Informative" denotes prompts that are neither *solve_all* nor *solve_none*.

Source difficulty	Composition	pass@1	pass@8	*solve_all*	*solve_none*	Informative
Easy→Easy	N	99.3	100.0	94.5	0.0	5.5
Easy→Easy	Y	89.2	98.6	69.9	1.4	28.8
Easy→Hard	N	66.3	78.3	51.7	21.7	26.7
Easy→Hard	Y	57.7	83.3	26.7	16.7	56.7
Hard→Easy	N	99.8	100.0	98.3	0.0	1.7
Hard→Easy	Y	68.5	93.1	34.5	6.9	58.6
Hard→Hard	N	81.2	94.0	64.2	6.0	29.9
Hard→Hard	Y	47.2	73.1	20.9	26.9	52.2

All evaluation codes are adapted from the DeepscaleR [3] codebase, and we use vLLM [69] to accelerate inference and Math-Verify to evaluate the LLMs' answers. For decoding, we follow [63] and set the temperature to 0.6, `top_p` to 0.95, `top_k` to 20, and the maximum output length to 32K tokens.

B.4 Details of Ablation for Candidate Sets $\mathcal{D}_k$

As noted in Section 6.1, the default configuration of *Composition-RL* sets $\mathcal{D}_2 = \mathcal{D}$ (the full prompt pool) and samples $\mathcal{D}_1$ as a small random subset ($|\mathcal{D}_2| = 12{,}000$, $|\mathcal{D}_1| = 20$). We also consider two variants: A) Both $\mathcal{D}_1$ and $\mathcal{D}_2$ are small randomly sampled subsets ($|\mathcal{D}_1| = |\mathcal{D}_2| = 500$). B) $\mathcal{D}_1$ is the full set $\mathcal{D}$, while $\mathcal{D}_2$ is a small randomly sampled subset ($|\mathcal{D}_1| = 12{,}000$, $|\mathcal{D}_2| = 20$). These settings are designed to yield roughly the same theoretical compositional dataset size.

After applying the filtering procedure in Section F.1, the resulting actual dataset sizes are:

- *Composition-RL*: 199K (see Section B.2).
- *Variant A*: 240K after step 1, 202K after step 2, and 200K after step 3.
- *Variant B*: 231K after step 1, 201K after step 2, and 200K after step 3.

Thus, the final dataset sizes are approximately matched across configurations.

C Analysis Details

In this appendix, we provide additional details for Section 6.2.

To evaluate compositional generalization, we use the same setting as in Section A.1. To determine whether the first variable v_1 is solved correctly, we prompt `Qwen2.5-32B-Instruct` using the default generation configuration and the prompt shown in Figure 4.

D More Analysis

D.1 Difficulty-Stratified Compositional Evaluation

We stratify the evaluation pairs by the difficulty of their first and second source problems. In Table 4, "N" and "Y" denote evaluation without and with composition, respectively. Composing the sources consistently reduces *solve_all* cases and increases informative cases, including when both sources are individually easy.

D.2 Overlap of Fully Solved Source Prompts

To reconcile the increasing fraction of fully solved source prompts with the continued utility of composition, we measure how often the two sources of a composed prompt are already *solve_all*. As training proceeds, most pairs contain at least one *solve_all* source, and by step 250 both sources are

Prompt for Verifying the Correctness of Finding v_1 in LLMs' Response

```
You are a math solution verifier.  Your task is to check if a given response
correctly solves for a specific intermediate variable in a composite math
problem.

**Problem:** {problem}

**Target Variable:** ${symbol}$
**Variable Definition:** {definition}
**Correct Answer for ${symbol}$:** {correct_answer}

**Model's Response:** {response}

--

**Your Task:** 1.  Identify what value the model computed for ${symbol}$ in its
response.
2.  Compare it with the correct answer:  {correct_answer}.
3.  Determine if the model correctly solved for ${symbol}$.

**Important Notes:** - Focus ONLY on whether ${symbol}$ was correctly computed;
ignore the final answer of the composite problem.
- The value might be stated explicitly (e.g., "${symbol}$ = 7") or implicitly
derived.
- Accept equivalent forms (e.g., "7", "7.0", "seven" are all correct if the
answer is 7).

**Output in JSON format:** {
"extracted_value":  "<the value the model gave for {symbol}, or 'NOT_FOUND' if
not mentioned>",
"is_equivalent":  <true if extracted_value equals {correct_answer}, false
otherwise>,
"reasoning":  "<brief explanation>",
"verdict":  "<CORRECT or INCORRECT>"
}
```

Figure 4: The Prompt for Verifying the Correctness of Finding v_1 in LLMs' Response.

Table 5: Overlap of *solve_all* source prompts during training (%).

Step	At least one source is *solve_all*	Both sources are *solve_all*
50	84.8	36.4
100	91.2	48.7
200	95.7	62.1
250	97.3	69.0

solve_all for 69.0% of pairs. Composition can therefore create informative training instances even from source prompts that have largely saturated in isolation.

Further evidence of implicit process supervision We provide more evidence of our "implicit process supervision" claim based on AIME24 using the 4B model. We consider three metrics: (i) **First-Attempt Correctness**, defined as the proportion of responses that produce the correct answer with a single `\boxed{}`; (ii) **Reflection Effectiveness**, defined as the accuracy conditioned on the presence of self-correction keywords; and (iii) **Self-Correction Success Rate**, defined as the correctness when the model revises its `\boxed{}` answer.

As shown in Table 6, our method consistently outperforms the baseline on all three metrics. Specifically, first-attempt correctness improves from 42.8 to 51.7, indicating that our method leads to stronger initial reasoning before any explicit revision. Moreover, the accuracy conditioned on reflection increases from 11.0 to 19.8, suggesting that reflective behavior is more effective under our method (implicit process supervision). Finally, the self-correction success rate rises substantially

Metric (%)	Baseline	Ours
First-attempt correctness	42.8	51.7
$P(\text{correct} \mid \text{reflection})$	11.0	19.8
Self-correction success	9.1	26.1

Table 6: Additional evidence of implicit process supervision on AIME24 with the 4B model.

Error Type	Original	Composition-RL
Linkage error	22	7
q_1 error	7	1
q_2 error	11	5

Table 7: Error analysis on 40 cases where the model solves both constituent questions in isolation but fails on the compositional question.

from 9.1 to 26.1, showing that when the model revises its answer, the revision is much more likely to be correct. Taken together, these results are consistent with implicit process supervision, with improvements in final accuracy, intermediate reasoning, and self-correction behavior.

Error analysis. We further conduct an error analysis on cases where the model can solve both constituent questions in isolation but fails on the corresponding compositional question. We manually inspect 40 such cases and categorize the errors into three types: (1) **Linkage error**, where the model fails at the "linkage condition" for any reasons; (2) q_1 **error**, where the model makes an error on q_1 when solving it in the compositional setting, despite being able to solve it correctly in isolation; and (3) q_2 **error**, where the model solves q_1 and derives the linkage correctly but still fails on q_2. As shown in Table 7, linkage error is the dominant failure mode for the original model, accounting for 22 out of 40 cases, compared with 7 q_1 errors and 11 q_2 errors. After applying *Composition-RL*, all three error types are reduced, with the largest reduction observed in linkage errors (from 22 to 7). Qualitatively, we find that linkage errors often arise from incorrect intermediate computation, unproductive self-verification loops, or selecting an incorrect linkage value despite otherwise correct reasoning. Interestingly, these examples also shed light on common failure modes of current reasoning models: (1) intermediate computation errors during the thinking process, (2) repeated self-verification loops that lead to truncation without progress, and (3) loss of direction mid-reasoning even when correct intermediate results are available. Composition-RL reduces linkage errors across all three failure modes, suggesting an improved ability to maintain correct intermediate reasoning throughout multi-step problems.

E More Results

E.1 About Average Response Length

In Section 5, we primarily discuss results based on `pass@1`; here, we also report the average response length in Table 8. As shown in Table 8, *Composition-RL* does not significantly increase response length across all model scales.

E.2 About `pass@k`

In Section 5, we primarily discuss results based on `pass@1`; here, we additionally report `pass@k` results in Table 9. As shown in Table 9, *Composition-RL* consistently improves `pass@k` across model scales, indicating that it broadens the set of solvable problems.

E.3 Variation Across Training Seeds

We repeat the 30B-A3B baseline and *Composition-RL* training runs with three random seeds. Table 10 reports the standard deviation across seeds; the corresponding mean scores are reported in Table 1. The small and comparable deviations indicate that the gains are stable across the evaluated seeds.

Table 8: Average response length (in K tokens) across different benchmarks. "ID Avg" denotes the average response length over in-domain mathematics benchmarks, "OOD Avg" denotes the average response length over out-of-domain multi-task benchmarks, and "Avg" denotes the overall average.

Composition	Mathematics (In-Domain)					Multi-Task (Out-Of-Domain)			Overall
	AIME 24	AIME 25	Beyond AIME	IMO-Answer	ID Avg	GPQA	MMLU-Pro	OOD Avg	Avg
Qwen3-4B-Base									
✘	16.0	13.4	13.3	12.8	13.9	4.2	2.3	3.3	10.3
✔	14.7	13.2	12.3	12.9	13.3	4.0	2.0	3.0	9.9
Qwen3-8B-Base									
✘	17.5	15.7	16.9	15.8	16.5	5.1	2.5	3.8	12.3
✔	14.2	14.9	15.9	14.9	15.0	5.5	2.8	4.2	11.4
Qwen3-14B-Base									
✘	13.6	13.4	12.9	12.1	13.0	3.7	1.9	2.8	9.6
✔	15.3	14.3	15.9	15.9	15.4	4.9	2.2	3.6	11.4
Qwen3-30B-A3B-Base									
✘	13.0	16.1	15.3	16.2	15.2	4.4	2.3	3.4	11.2
✔	13.1	15.6	14.5	15.3	14.6	4.3	2.1	3.2	10.8

Table 9: Pass@k (%) across different benchmarks. The value of k for each benchmark is consistent with that used in the corresponding Avg@k setting in the main results table. "ID Avg" denotes the average pass@k over in-domain mathematics benchmarks, "OOD Avg" denotes the average pass@k over out-of-domain multi-task benchmarks, and "Avg" denotes the overall average.

Composition	Mathematics (In-Domain)					Multi-Task (Out-Of-Domain)			Overall
	AIME 24 Pass@32	AIME 25 Pass@32	Beyond AIME Pass@8	IMO-Answer Pass@4	ID Avg Avg.	GPQA Pass@8	MMLU-Pro Pass@1	OOD Avg Avg.	Avg Avg.
Qwen3-4B-Base									
✘	56.7	46.7	23.0	25.5	38.0	79.3	58.6	69.0	48.3
✔	66.7	56.7	31.0	26.0	45.1	77.8	61.4	69.6	53.3
Qwen3-8B-Base									
✘	66.7	46.7	29.0	30.7	43.3	81.3	62.6	72.0	52.8
✔	73.3	53.3	35.0	32.0	48.4	81.3	64.5	72.9	56.6
Qwen3-14B-Base									
✘	70.0	70.0	39.0	34.5	53.4	81.8	67.2	74.5	60.4
✔	76.7	76.7	39.0	41.5	58.5	81.8	69.3	75.6	64.2
Qwen3-30B-A3B-Base									
✘	73.3	56.7	39.0	40.0	52.3	83.8	62.6	73.2	59.2
✔	76.7	63.3	40.0	39.0	54.8	84.8	64.6	74.7	61.4

E.4 Generalization to a Non-Qwen Backbone

We additionally evaluate the baseline and *Composition-RL* with the non-Qwen `OctoThinker-3B` backbone [70]. As shown in Table 11, *Composition-RL* improves all nine benchmarks, indicating that its benefits are not specific to the Qwen model family.

E.5 Ratio of *solve_all* and *solve_none*

We also report the proportions of *solve_all* and *solve_none* examples across training steps in Figure 5. As shown in Figure 5, at later stages of training, *Composition-RL* exhibits a lower *solve_all* ratio and a comparable *solve_none* ratio. Therefore, *Composition-RL* yields a larger amount of effective training data overall.

F Details of *SPC*

F.1 Reliability of *SPC*

The full *SPC* pipeline can be automated with an LLM assistant; implementation details and the corresponding prompts are provided in Section F.2 and Section F.4, respectively. To make this automated process more reliable, we have some additional verification steps to filter potential mistakes during composing. Following [26], we use LLM-based self-verification at each composition step. Concretely, we prompt the same LLM to perform the following checks:

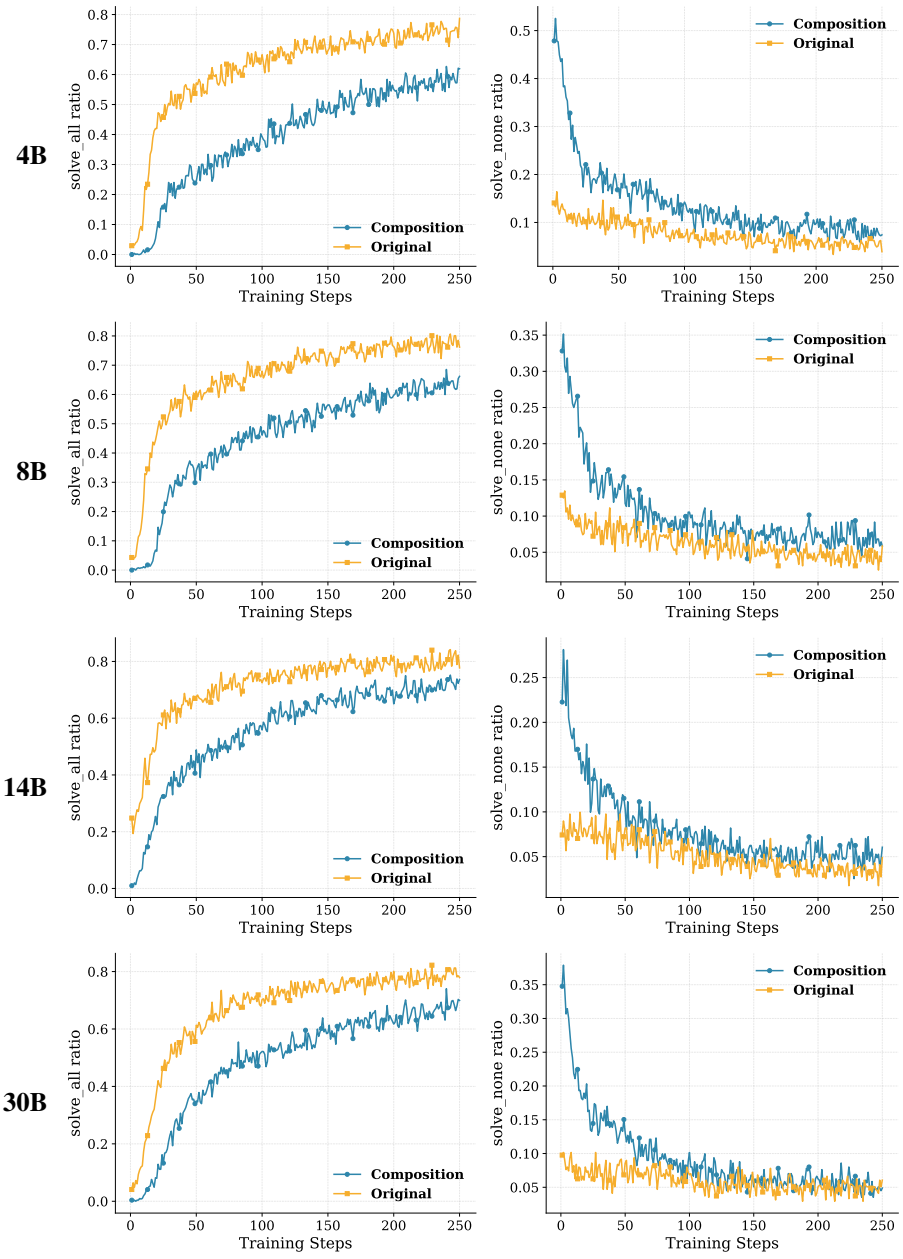

Figure 5: *solve_all* and *solve_none* ratio across model sizes.

Table 10: Standard deviation of `pass@1` (percentage points) over three training seeds for `Qwen3-30B-A3B-Base`.

Benchmark	Baseline	*Composition-RL*
AIME24	1.58	1.65
AIME25	1.52	1.60
BeyondAIME	1.23	1.35
GPQA	1.08	1.14
IMOBench	1.27	1.33
MMLU-Pro	0.80	0.92

Table 11: `pass@1` (%) with `OctoThinker-3B`.

Benchmark	Baseline	*Composition-RL*
AIME24	3.44	5.94
AIME25	1.88	2.81
AMC	24.85	28.12
BeyondAIME	1.25	1.62
GPQA	25.88	29.23
IMO-Answer	5.31	5.44
MATH	57.60	63.15
MMLU-Pro	34.55	36.66
OlympiadBench	21.33	22.81

- ❶ **Verification of "Modify q_1 with gt_1."** In this step, the LLM extracts a variable v_1 from gt_1 and its definition d_1. We then ask the LLM to compute the value of v_1 given q_1 and d_1, and compare the computed value against the extracted v_1. If they do not match, we discard the prompt. This verification improves the reliability of the modification of q_1.

- ❷ **Verification of "Modify q_2."** Analogously, we prompt the LLM to verify whether the extracted variable v_2 (and its definition) is consistent with q_2. Prompts that fail this check are filtered out.

- ❸ **Verification of "Connect q_1 and q_2."** This step primarily involves concatenation. To ensure quality, we prompt the LLM to check for inconsistencies (e.g., conflicting variable names) and filter out any inconsistent prompts.

This verification procedure removes many low-quality compositions, leaving a substantially more reliable set of composed prompts. As reported in [26], the rate of erroneous prompts after filtering is below 2%. We believe this error rate is acceptable for training.

F.2 Implementation Details

We use `Qwen2.5-32B-Instruct` [67] with step-specific prompts to implement each stage of Section 4.1 as well as the verification procedure in Section F.1. Unless otherwise specified, we set the temperature to 0.1, `top_p` to 0.7, and the maximum output length to 4096 tokens. The prompts are provided in Section F.4.

F.3 Physics-to-Mathematics Composition Examples

The following examples give both source questions and the complete composed prompt. The answer-derived value from the physics question q_1 determines a replacement in the mathematics question q_2.

Example 1: Mechanics to Complex Numbers

First question q_1: If a force of $50\,\mathrm{N}$ is applied to an object with an acceleration of $2\,\mathrm{m\,s^{-2}}$, what is the mass of the object? Its ground-truth answer is $25\,\mathrm{kg}$.

Second question q_2: Let z_1, z_2, z_3 be complex numbers such that $|z_1| = 2$, $|z_2| = 3$, and $|z_3| = 4$. Find the largest possible value of $|z_1 - z_2|^2 + |z_1 - z_3|^2 + |z_2 - z_3|^2$. Its ground-truth answer is 87.

Composed question $q_{1:2}$: If a force of $50\,\mathrm{N}$ is applied to an object with an acceleration of $2\,\mathrm{m\,s^{-2}}$, what is the mass of the object? Let X denote the numerical value of the mass in kilograms, and let n be 23 less than X. Let z_1, z_2, z_3 be complex numbers such that $|z_1| = n$, $|z_2| = 3$, and $|z_3| = 4$. Find the largest possible value of $|z_1 - z_2|^2 + |z_1 - z_3|^2 + |z_2 - z_3|^2$. Here $X = 25$ and $n = 2$, so the composed prompt preserves the ground-truth answer 87.

Example 2: Magnetism to Complex Roots

First question q_1: An air-cored coil has a flux density of $25\,\mathrm{mT}$. When an iron core is inserted, the flux density increases to $1.6\,\mathrm{T}$. Calculate the relative permeability of the iron. Its ground-truth answer is $\mu_r = 64$.

Second question q_2: Let x be a complex number such that $x^{2011} = 1$ and $x \neq 1$. Compute $\frac{x^2}{x-1} + \frac{x^4}{x^2-1} + \frac{x^6}{x^3-1} + \cdots + \frac{x^{4020}}{x^{2010}-1}$. Its ground-truth answer is 1004.

Composed question $q_{1:2}$: An air-cored coil has a flux density of $25\,\mathrm{mT}$. When an iron core is inserted, the flux density increases to $1.6\,\mathrm{T}$. Calculate the relative permeability μ_r of the iron, and let κ be 63 less than μ_r. Let x be a complex number such that $x^{2011} = \kappa$ and $x \neq \kappa$. Compute $\frac{x^2}{x-\kappa} + \frac{x^4}{x^2-\kappa} + \frac{x^6}{x^3-\kappa} + \cdots + \frac{x^{4020}}{x^{2010}-\kappa}$. Here $\mu_r = 64$ and $\kappa = 1$, so the composed prompt preserves the ground-truth answer 1004.

F.4 Prompts of *SPC*

Following [26], we provide the prompt used to modify q_1 in 6 and the self-verification prompt used to check the modification in 7. We use similar prompts for the other steps of *SPC*, and the complete prompt set is available in our released code.

Prompt for Modifying q_1 with gt_1

Given a math problem and the final answer, your task is to find out one number from the answer and provide the corresponding definition. Follow the steps below:
Step 1: Identify a specific integer, float, or fraction within `final_answer` and name it as $new_variable1$; There are several situations:

1. If the `final_answer` contains unknown variables:
   (a) If the `final_answer` is an expression, choose one coefficient as $new_variable1$, for example, $2x + 3$, you can choose the coefficient of x as $new_variable1$, which is 2, and in the case of $\sin(x)$, there is a hidden coefficient 1 and a hidden amplitude 1, you can choose either one as $new_variable1$;
   (b) If the `final_answer` is an equation, you can choose one solution as $new_variable1$, for example, $y = 2x + 1$, you can define the value of y as $new_variable1$ when given $x = 1$, which is 3;
   (c) If the `final_answer` is a symbol of an option or a word, such as 'A', 'B', 'CAT', etc., use their first letter's order in the alphabet as a variable, such as 'A' = 1, 'B' = 2, 'CAT' = 3, etc.;
   (d) If the `final_answer` contains 2 or more items, e.g., multiple choice questions, choose the smallest or the largest one, and then apply the corresponding situation.
2. If the `final_answer` has no unknown variables, there are several situations:
   (a) If the `final_answer` itself is a numerical value, like 'four', '4', '$2 + \sqrt{2}$', '3π', and '$\frac{3}{4}$', use it directly as $new_variable1$;
   (b) If the `final_answer` contains 2 or more numerical values, use the largest or the smallest one as $new_variable1$;
   (c) If the `final_answer` is an interval or ratio, choose one boundary and ∞ is not allowed, for example, $[2, \infty)$, you can define the lower bound as $new_variable1$, which is 2;
   (d) If the `final_answer` is a ratio, choose one part of the ratio, for example, $3 : 4$; you can define the first part of the simplified ratio as $new_variable1$, which is 3;
   (e) If the `final_answer` is a non-base 10 number, for example, 1001_2, you can define 'the number of digits in the base 2 representation' as $new_variable1$, which is 4;
   (f) If the `final_answer` is an angle or degree, choose the corresponding radian value, for example, 30° or $30°$, define the corresponding radian value of final answer as $new_variable1$, which is $\pi/6$.

All in all, find a way to identify a specific numerical value as $new_variable1$ without unknown, and make sure the reader can get the value of $new_variable1$ from the `final_answer` through your definition.
Step 2: Output the value of $new_variable1$, keep the exact value or math symbol, and simplify the fraction if necessary, for example, keep π as π, keep $\sqrt{2}$ as $\sqrt{2}$, and simplify $\frac{6}{8}$ as $\frac{3}{4}$, without rounding to a decimal point.
Step 3: Output the definition of $new_variable1$ without mentioning the real value.

—

Output Format: ... (omit for simplicity)
Examples: ... (omit for simplicity)

Figure 6: The Prompt for Generating Variable v_1 and Definition d_1 for q_1.

Prompt for Verifying the Modification of q_1

1. Check the extraction of variable v_1:
`{Problem 1}`
Assume that the final answer of the problem is `{FINAL_ANSWER}`. `{DEFINITION_OF_NEW_VARIABLE1}`
Then what is the value of $new_variable1$?
Please output the value of $new_variable1$ directly, wrapping it in `\boxed{}`, for example, `\boxed{3}`.

2. Check the value of v_1 using Python:
Task Description:
Write a Python program to compare two given values and determine if they are equal. Follow these guidelines:

1. Use the `sympy` library to handle symbolic comparisons, ensuring that equivalent expressions (e.g., $\frac{2}{4}$ and $\frac{1}{2}$) are recognized as equal.
2. For values involving irrational constants (e.g., π, e), perform comparisons up to **two decimal places** for practical equivalence.
3. Include clear intermediate steps in the program, such as evaluating or simplifying the values where appropriate.
4. Wrap the final comparison outcome in a `\boxed{}` command for clarity.
5. Provide both the Python code and the results of running the code.

Output Format:

```
```python
{The Python code that compares the two given values, including print statements
for intermediate steps and the \boxed{final comparison outcome}.}
```

```output
{The output of the Python program.}
```

[Examples Here]
```

Figure 7: The Prompt for Verifying the Modification of q_1.

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: Our abstract and introduction summarize our scope and contribution.

   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: We include a Limitation Section at the beginning of the Appendix.

   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: [N/A]

Justification: This paper does not include theoretical results.

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: We clearly disclose our experimental setting in Section 5.1 and Section B.

   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: Our code, data, and models are publicly available.

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: We clearly disclose our experimental setting in Section 5.1 and Section B.

   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: [No]

   Justification: As stated in Section 5.1 and Section B, we follow the standard recipe to report `pass@1`.

   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: Partially yes, all the training details are given in Section B.1.

   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: We adhere to the NeurIPS Code of Ethics.

   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: [Yes]

    Justification: We have included an "Impact Statement" section in the appendix.

    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: This paper poses no such risks

    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: [Yes]

    Justification: We clearly cited all existing assets in Section 5.1 and Section B.

    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 details of generating new assets are given in Section F.

    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: This paper does not involve crowdsourcing nor research with human subjects.

    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: This paper does not involve crowdsourcing nor research with human subjects.

    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: We include a "Declaration of LLM Usage" section to discuss LLM use.

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.