Self-Hinting Language Models Enhance Reinforcement Learning

Baohao Liao [*,†,1,2] **Hanze Dong** [*,1] **Xinxing Xu** [1] **Christof Monz** [2] **Jiang Bian** [1]

Abstract

Group Relative Policy Optimization (GRPO) has recently emerged as a practical recipe for aligning large language models with verifiable objectives. However, under sparse terminal rewards, GRPO often stalls because rollouts within a group frequently receive identical rewards, causing relative advantages to collapse and updates to vanish. We propose *self-hint aligned GRPO with privileged supervision* (SAGE), an on-policy reinforcement learning framework that injects privileged hints during training to reshape the rollout distribution under the *same* terminal verifier reward. For each prompt x, the model samples a compact hint h (e.g., a plan or decomposition) and then generates a solution τ conditioned on (x, h). Crucially, the task reward $R(x, \tau)$ is unchanged; hints only increase within-group outcome diversity under finite sampling, preventing GRPO advantages from collapsing under sparse rewards. At test time, we set $h = \varnothing$ and deploy the no-hint policy without any privileged information. Moreover, sampling diverse self-hints serves as an adaptive curriculum that tracks the learner's bottlenecks more effectively than fixed hints from an initial policy or a stronger external model. Experiments over 6 benchmarks with 3 LLMs show that SAGE consistently outperforms GRPO, on average +2.0 on Llama-3.2-3B-Instruct, +1.2 on Qwen2.5-7B-Instruct and +1.3 on Qwen3-4B-Instruct. The code is available at https://github.com/BaohaoLiao/SAGE.

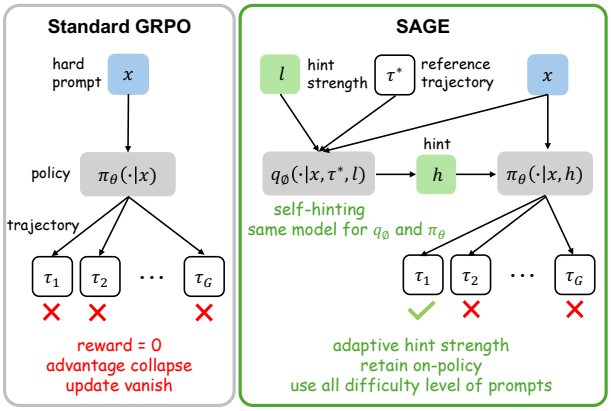

Figure 1. An overview of our proposed method, SAGE. When an LLM can't sample any correct trajectory for a prompt, the LLM **self-generates** hint from the reference solution of the prompt. The hint is then used together with the difficult prompt as input to the LLM, avoiding advantage collapse and ensuring the sampling of correct trajectories to update the policy model.

1. Introduction

Reinforcement learning (RL) has become a core tool for training and aligning large language models (LLMs), particularly when supervision is most naturally expressed via verifiable objectives such as exact-match correctness, unit tests, or automated checkers (Ouyang et al., 2022; Schulman et al., 2017; DeepSeek-AI et al., 2025). In this setting, the objective is straightforward: maximization the expected reward over prompts, yet optimization can be fragile: with finite sampling, policy-gradient estimators may exhibit high variance and can even become degenerate on hard prompts.

A salient example arises with Group Relative Policy Optimization (GRPO) (Shao et al., 2024) under sparse terminal rewards. GRPO centers (and often standardizes) rewards within each rollout group, relying on within-group outcome differences to produce a nonzero update. With a $0/1$ verifier, difficult prompts frequently yield groups where all rollouts receive the same reward (typically all zeros). In that case, the group-centered advantages collapse and the minibatch policy-gradient estimate becomes identically zero. Importantly, this is a **finite-sample pathology**: the underlying expected objective needs not be flat, but the estimator provides no learning signal for many prompts.

Existing remedies largely modify data collection. A common baseline is to skip uninformative updates (e.g., degenerate groups) and resample prompts, which improves performance but implicitly biases training toward easier prompts (Yu et al., 2025; Xiong et al., 2025a). More systematic

[*]Equal contribution [†]This work was done during an internship at Microsoft. [1]Microsoft Research [2]Language Technology Lab, University of Amsterdam. Correspondence to: Hanze Dong <hanzedong@microsoft.com>.

Preprint. February 3, 2026.

approaches include adaptive sampling or curriculum-style scheduling to allocate more rollouts to difficult prompts (Yao et al., 2025; Xiong et al., 2025b; Li et al., 2025; Zhang et al., 2025c), as well as leveraging offline data or externally generated candidates (e.g., from stronger models) to bootstrap learning (Zhang et al., 2025a; Yan et al., 2025; Zhang et al., 2025b). While effective, these strategies can either skew the training distribution or introduce context/distribution mismatch that must be handled carefully.

We propose SAGE (Self-hint Aligned GRPO with Privileged Supervision), a complementary approach based on *privileged hinting*. During training, we provide an additional hint h, a lossy compression of a reference solution $\tau^\star$, and roll out from the hint-conditioned policy $\pi_\theta(\cdot \mid x, h)$. Hints only reshape the rollout distribution to increase the probability of observing mixed outcomes within a finite group. At test time, we deploy the no-hint policy. We refer to hints generated by the policy itself as self-hints, and to the procedure of generating such hints as self-hinting.

This degeneracy can be made explicit. Let $p_\theta(x)$ be the no-hint success probability and G the group size. The probability that a rollout group contains mixed outcomes is $1-(1-p_\theta(x))^G - p_\theta(x)^G \approx Gp_\theta(x)$, so updates vanish whenever $Gp_\theta(x) \ll 1$. Hinting is useful precisely when it increases the effective success probability so that mixed-outcome groups become common for the same G.

Contributions. (i) We introduce SAGE as shown in Figure 1, an on-policy RL framework that conditions rollouts on privileged hints during training while keeping the task reward unchanged and removing hints at test time. (ii) We develop a policy-dependent hint-strength scheduler, that activates hints only when within-group rewards collapse, yielding an automatic curriculum. (iii) We propose an online self-hinting scheme that periodically refreshes the hint distribution during training to maintain calibration to the learner, avoiding overly weak/overly strong fixed hints. (iv) We provide analysis that characterizes GRPO collapse as a gate-opening probability under Bernoulli rewards and empirically validate that SAGE improves sample efficiency and final accuracy on challenging reasoning benchmarks.

2. RL with Privileged Hinting

Vanilla GRPO works well when a prompt x yields occasional positive rollouts. In hard regimes, groups often receive identical rewards, collapsing within-group advantages and stalling learning. We address this failure mode by injecting *privileged hints* during training that keep the reward unchanged while reshaping the rollout distribution to surface informative trajectories under finite sampling.

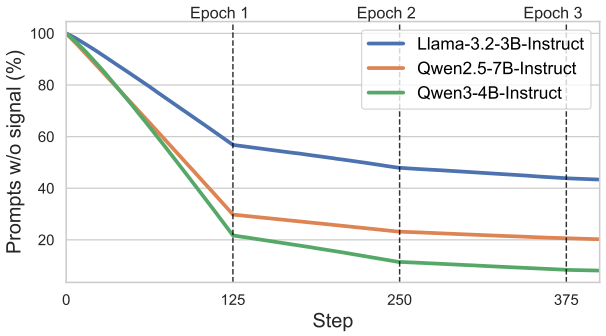

Figure 2. The percentage of prompts whose correct trajectories have NEVER been sampled w.r.t. the training step. Here we train on 64k prompts, and sample 8 traces per prompt per step. A large number of prompts is wasted during RL, especially for a weaker LLM, since they don't offer any signal for training.

2.1. Setup and the GRPO stall

Let $x \sim \mathcal{D}$ be a prompt. A policy π_θ generates a trajectory $\tau = (y_1, \ldots, y_T)$, written as $\tau \sim \pi_\theta(\cdot \mid x)$. We use a binary reward $R(x,\tau) \in \{0,1\}$. Define the success probability

$$p_\theta(x) = \Pr_{\tau\sim\pi_\theta(\cdot|x)}[R(x,\tau)=1]. \tag{1}$$

GRPO implementations often standardize groupwise advantages by the within-group standard deviation. For a group of G rollouts $\{\tau_i\}_{i=1}^G$, let

$$R_i = R(x,\tau_i),\ \bar{R} = \tfrac{1}{G}\sum_{i=1}^G R_i,\ s^2 = \tfrac{1}{G}\sum_{i=1}^G (R_i-\bar{R})^2,$$

and define standardized advantages

$$A_i = \frac{R_i - \bar{R}}{s+\epsilon}, \tag{2}$$

where $\epsilon \geq 0$ is a numerical stabilizer and $R_i \in \{0,1\}$. When $p_\theta(x)$ is tiny, a group is often all-zero and $A_i = 0$ for all i. The chance of a non-degenerate group is

$$1-(1-p_\theta(x))^G - p_\theta(x)^G \approx 1-(1-p_\theta(x))^G \approx Gp_\theta(x),$$

so training stalls whenever $Gp_\theta(x) \ll 1$ on most prompts. Figure 2 illustrates this phenomenon in practice: for many hard prompts, correct trajectories are never sampled for a long stretch of training, yielding no learning signal.

2.2. Privileged hinting as sampling

When a reference trajectory $\tau^\star$ is available during training, we generate a hint h as a lossy compression of $\tau^\star$. The hint is appended to the prompt as additional context.

Hint strength and the no-hint case. We control hint informativeness with a discrete strength level $\ell \in \{0,1,\ldots,L\}$,

where larger ℓ indicates more information about the reference trajectory $\tau^\star$. We sample

$$\ell \sim p(\ell), \qquad h \sim q(h \mid x, \tau^\star, \ell), \tag{3}$$

where $\ell = 0$ corresponds to the no-hint setting and $q(h \mid x, \tau^\star, 0) = \delta_\varnothing(h)$ (i.e., $h = \varnothing$ deterministically).

Policy-dependent scheduling of ℓ. Hints should be used only when a prompt provides no learning signal. We let the sampling of ℓ depend on the policy through a simple statistic, such as a collapse indicator $c(x) = \mathbb{I}\big[\mathrm{Var}\big(\{R(x,\tau_i)\}_{i=1}^G\big) = 0\big]$, computed from a small probe group under the policy model π_θ. A minimal scheduler is

$$p(\ell \mid x) = \begin{cases} \delta_0, & c(x) = 0, \\ p(\ell), & c(x) = 1, \end{cases} \tag{4}$$

so $\ell > 0$ is activated only when the group collapses.

With a hint, we sample from $\pi_\theta(\cdot \mid x, h)$. The success rate increases: $p_\theta^{(\ell)}(x) = \Pr_{\tau\sim\pi_\theta(\cdot|x,h)}[R(x,\tau) = 1], h \sim q(\cdot \mid x, \tau^\star, \ell)$. Hinting is useful when it raises $p_\theta^{(\ell)}(x)$ enough that $Gp_\theta^{(\ell)}(x)$ is no longer tiny, so non-degenerate groups become common and RL receives updates.

2.3. GRPO with hints and the final loss

Given x, sample ℓ and h, then draw a group of rollouts

$$\tau_i \sim \pi_\theta(\cdot \mid x, h), \qquad R_i = R(x, \tau_i), \quad i = 1, \ldots, G.$$

Compute A_i with Eq. (2). The loss conditioned on hints is

$$\begin{aligned}\mathcal{L}(\theta) = -\mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G} A_i \sum_{t=1}^{T_i} \log \pi_\theta(y_{i,t} \mid x, h, y_{i,<t})\right] \\ +\beta\, \mathbb{E}\big[\mathrm{KL}\big(\pi_\theta(\cdot \mid x, h) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid x, h)\big)\big],\end{aligned} \tag{5}$$

where the expectation is over $x \sim \mathcal{D}$, the hint sampling from Eq. (3), and rollouts from $\pi_\theta(\cdot \mid x, h)$. At test time we set $\ell = 0$, $h = \varnothing$ and run the policy $\pi_\theta(\cdot \mid x, \varnothing) \equiv \pi_\theta(\cdot \mid x)$.

Summary. Sparse rewards can cause GRPO to stall because, for many prompts x, finite groups contain no positive samples and advantages collapse. Privileged hinting fixes this by changing the rollout distribution for such prompts while keeping the reward unchanged. A policy-dependent scheduler activates hints only when groups collapse, yielding an automatic curriculum. Training remains on-policy since rollouts are drawn from $\pi_\theta(\cdot \mid x, h)$. Deployment uses $\ell = 0$ and requires no hints or privileged information.

3. Analysis

3.1. Standardized GRPO as a gated update objective

Fix a context (x, h) and draw G rollouts with rewards $R_i \in \{0, 1\}$. Let $\bar{R} = \frac{1}{G}\sum_i R_i$ and $s^2 = \frac{1}{G}\sum_i (R_i - \bar{R})^2$. Standardized GRPO uses $A_i = \frac{R_i - \bar{R}}{s+\epsilon}$.

Corollary 3.1 (Signal energy equals a gate probability)**.** *Define the advantage energy $E \coloneqq \frac{1}{G}\sum_{i=1}^G A_i^2$. If $\epsilon > 0$,*

$$E = \frac{s^2}{(s+\epsilon)^2} \in [0, 1], \tag{6}$$

which is monotone in s and still collapses to 0 when $s = 0$.

For GRPO, the prompt-level update magnitude is dominated by whether the group is *non-degenerate* ($s > 0$). In other words, training behaves like a gated procedure that updates only when the rollout group contains mixed outcomes.

Proposition 3.2 (Gate opening probability under Bernoulli rewards)**.** *Let $p_\theta(x, h) = \Pr[R(x, \tau) = 1 \mid x, h]$. Then*

$$\Pr[s > 0 \mid x, h] = 1 - (1 - p_\theta(x, h))^G - p_\theta(x, h)^G, \tag{7}$$

$\Pr[s > 0 \mid x, h]$ is maximized at $p_\theta = \frac{1}{2}$. In the sparse regime $p_\theta(x, h) \ll 1$, $\Pr[s > 0 \mid x, h] \approx G\, p_\theta(x, h)$.

Thus, SAGE should choose the hint strength to move hard prompts out of the regime where $Gp_\theta \ll 1$, and avoid overly strong hints that push $p_\theta \approx 1$ to close the gate again.

Proposition 3.3 (Optimal hint distribution is policy-dependent)**.** *Fix a prompt x and group size $G \geq 2$. Let $p_\theta(x, h) = \Pr_{\tau\sim\pi_\theta(\cdot|x,h)}[R(x, \tau) = 1]$ and define*

$$u(p) \coloneqq 1 - (1 - p)^G - p^G. \tag{8}$$

Under Bernoulli rewards, $u(p_\theta(x, h)) = \Pr[s > 0 \mid x, h]$.

For any distribution $q(\cdot \mid x)$, define the expected probability

$$J_x(\theta, q) \coloneqq \mathbb{E}_{h\sim q(\cdot|x)}[u(p_\theta(x, h))]\,. \tag{9}$$

Then u is symmetric and strictly concave on $[0, 1]$, and is maximized at $p = \frac{1}{2}$. Consequently, any maximizer

$$q_\theta^\star(\cdot \mid x) \in \arg\max_q J_x(\theta, q) \tag{10}$$

must place its mass on calibrating hints *that make $p_\theta \approx \frac{1}{2}$.*

In general, the set of calibrating hints depends on θ. Unless $p_\theta(x, h)$ is invariant in θ for q-almost all h, a fixed q cannot remain (near) optimal for $J_x(\theta, q)$ throughout training. Updating q online therefore reduces this gate-mismatch and increases the frequency of non-degenerate GRPO updates.

Remark 3.4 (Why not sample many hints per prompt.). Let $u(p) = 1 - (1 - p)^G - p^G$ denote the gate probability in (7). For $G \geq 2$, u is concave on $[0, 1]$ since $u''(p) = -G(G-1)\big(p^{G-2} + (1-p)^{G-2}\big) \leq 0$. Therefore, for any hint distribution $h \sim q$,

$$\mathbb{E}_{h\sim q}\big[u(p_\theta(x, h))\big] \leq u(\mathbb{E}_{h\sim q}[p_\theta(x, h)])\,. \tag{11}$$

At a fixed mean success rate, additional randomness across hints can only reduce the expected frequency of non-degenerate groups. Motivated by (11), we sample a single hint realization per prompt per epoch and spend compute on G or better strength scheduling.

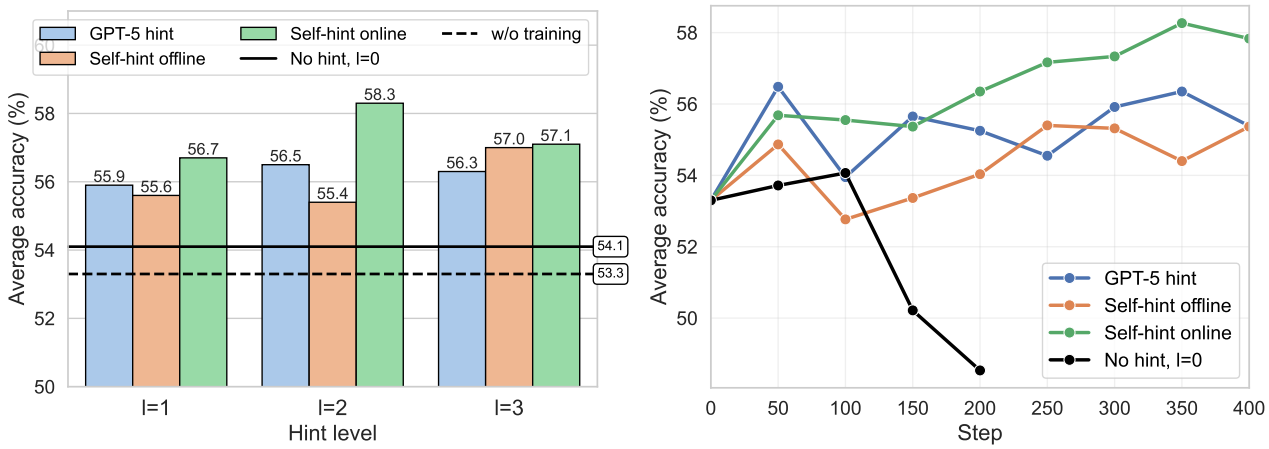

Figure 3. Average accuracy on Qwen3-4B-Instruct over 6 benchmarks. The 4.5k training prompts here are extremely hard, whose correct trajectories have never been sampled during training as Figure 2. The number of rollouts per prompt per step here is set to 32 to encourage exploration. **Left**: Performance on various hints. Training without hint only slightly improves the performance, since the reward signal from the hard prompts is sparse. However, training with any hint boosts the performance. Among all methods, online self-hinting consistently achieves the best performance across different hint levels. **Right**: Average accuracy w.r.t. the training steps for hint level $l = 2$. Training without any hint even degrades the performance as the training goes, since the reward signal is too sparse, making it overfit to a few solvable prompts. However, online self-hinting boosts the performance steadily. Refer to Table C.1 for detailed number.

Overall, standardized GRPO turns sparse-reward learning into maximizing a gate probability. The proposed method should operationalize this by (i) ensuring on-policy conditioning on h, (ii) scheduling ℓ when the gate is closed, (iii) updating the self-hint generator online to keep p_θ calibrated.

4. Design of SAGE

4.1. On-policy training

Why hints must be in the conditioning context. SAGE appends a hint h to the prompt and samples rollouts from the hint-conditioned policy $\pi_\theta(\cdot \mid x, h)$. This is not merely a modeling choice: it is what keeps training *on-policy* for the augmented context. The loss is $\sum_t \log \pi_\theta(y_t \mid x, h, y_{<t})$. If one instead samples $\tau \sim \pi_\theta(\cdot \mid x, h)$ but evaluates $\log \pi_\theta(\tau \mid x)$ (i.e., dropping h inside the log-prob), the update no longer corresponds to the gradient of any on-policy objective under the sampling process. In practice this mismatch behaves like an off-policy update and is markedly less stable under sparse rewards. We also include a controlled ablation (Sec. 5.2) that keeps the *sampling* process identical (roll out with hint) but changes the conditioning.

4.2. Online self-hinting

In the algorithm, we can produce privileged hints in two ways: (1) **Offline hints (fixed).** A fixed hint generator (e.g., extracted once from $\tau^\star$) is simple, but it does not adapt to the learner and can become miscalibrated over training. (2) **Online hints.** We periodically refresh the hint generator using a copy of the current policy π_θ.

In SAGE, hint generation is *online*. We implement $q_\phi(h \mid x, \tau^\star, \ell)$ by prompting the policy π_θ to produce a *procedure-only* plan aligned with the reference trajectory.

We evaluate three variants: (1) **Fixed privileged hints:** q_ϕ is derived from π_{θ_0} and frozen after initialization. (2) **Online privileged hints (SAGE):** q_ϕ is derived from π_θ and refreshed during training. (3) **External teacher hints:** q_ϕ is produced by a stronger frozen model, when available.

By Figure 3, adding hints improves performance across hint levels compared with no hint, consistent with hints increasing the chance of sampling informative trajectories under sparse rewards. Besides, online self-hinting consistently performs best, indicating that continually refreshed self-hints are better calibrated to the learner than fixed hints.

4.3. Policy-dependent scheduling

We control hint informativeness with a discrete strength variable $\ell \in \{0, 1, \ldots, L\}$, where $\ell = 0$ corresponds to the deployable no-hint setting. The scheduler is *policy-dependent* in the sense that it adapts ℓ using statistics collected from recent rollouts under the policy π_θ (stop-gradient). Intuitively, we increase ℓ only when the current policy provides insufficient learning signal on a prompt.

Scheme 1 (SAGE-LIGHT): epoch-level accuracy threshold. Let $\hat{p}_{t-1}(x)$ denote the empirical success rate of prompt x measured in the previous epoch using rollouts from policy model with current hint level. Given a target threshold $\alpha \in (0, 1)$, we increase hint strength when the prompt is too

Algorithm 1 SAGE / SAGE-LIGHT: Self-hint Aligned GRPO with Privileged Supervision

Require: Training set $\mathcal{D} = \{(x, \tau^\star)\}$, policy model π_θ, group size G, KL weight β, stabilizer $\epsilon > 0$, max hint level L, reference policy π_{ref}, hint generator $q_\phi(h \mid x, \tau^\star, \ell)$ based on π_θ, threshold α (SAGE-LIGHT only).

Initialize policy parameters θ; initialize per-prompt level map $\ell(x) \leftarrow 0$ for all $x \in \mathcal{D}$.
Repeat for epochs:
  for each minibatch $\{(x_b, \tau_b^\star)\}_{b=1}^B$:
    for $b = 1, \ldots, B$ **do**
      if SAGE-LIGHT **then**
        if epoch > 1 **and** $\bar{R}_b < \alpha$ **then** $\ell(x_b) \leftarrow \min\{\ell(x_b) + 1, L\}$
        Sample $h_b \sim q_\phi(h \mid x_b, \tau_b^\star, \ell(x_b))$
      else (SAGE)
        for $\ell = 0, \ldots, L$ **do**
          Sample $\tilde{h}_b \sim q_\phi(h \mid x_b, \tau_b^\star, \ell)$
          Sample $\tilde{\tau}_{b,i} \sim \pi_\theta(\cdot \mid x_b, \tilde{h}_b)$ for $i = 1, \ldots, G$ and compute $\tilde{R}_{b,i} \leftarrow R(x_b, \tilde{\tau}_{b,i})$
          if $\sum_{i=1}^G \tilde{R}_{b,i} > 0$ **or** $\ell = L$ **then**
            $h_b \leftarrow \tilde{h}_b,\ \tau_{b,i} \leftarrow \tilde{\tau}_{b,i},\ R_{b,i} \leftarrow \tilde{R}_{b,i}$; **break**
      (If SAGE-LIGHT) Sample $\tau_{b,i} \sim \pi_\theta(\cdot \mid x_b, h_b)$ for $i = 1, \ldots, G$ and compute $R_{b,i} \leftarrow R(x_b, \tau_{b,i})$.
      Compute $\bar{R}_b \leftarrow \frac{1}{G}\sum_{i=1}^G R_{b,i}$, $s_b \leftarrow \sqrt{\frac{1}{G}\sum_{i=1}^G (R_{b,i} - \bar{R}_b)^2}$, $A_{b,i} \leftarrow \frac{R_{b,i} - \bar{R}_b}{s_b + \epsilon}$.
    $\mathcal{L}_{\text{pg}} \leftarrow -\frac{1}{BG}\sum_{b=1}^B \sum_{i=1}^G A_{b,i} \sum_{t=1}^{T_{b,i}} \log \pi_\theta(y_{b,i,t} \mid x_b, h_b, y_{b,i,<t})$
    $\mathcal{L}_{\text{kl}} \leftarrow \frac{1}{B}\sum_{b=1}^B \text{KL}(\pi_\theta(\cdot \mid x_b, h_b) \,\|\, \pi_{\text{ref}}(\cdot \mid x_b, h_b))$
    Update $\theta \leftarrow \theta - \eta \nabla_\theta \big(\mathcal{L}_{\text{pg}} + \beta \mathcal{L}_{\text{kl}}\big)$
Deployment: set $\ell = 0$ so $h = \varnothing$, and run $\pi_\theta(\cdot \mid x)$.

hard:

$$\ell_t(x) = \min\{\ \ell_{t-1}(x) + 1,\ L\ \} \quad \text{if } \hat{p}_{t-1}(x) < \alpha, \quad (12)$$

and otherwise keep $\ell_t(x) = \ell_{t-1}(x)$. Thus, hints are activated only when success is consistently rare.

Scheme 2 (SAGE): group-degeneracy trigger. GRPO requires within-group outcome differences to produce a nonzero update. We therefore use a more local trigger based on whether a probe group contains any positive sample. For a small probe group $\{\tau_i\}_{i=1}^G$ rolled out from $\pi_\theta(\cdot \mid x, h)$ at the current strength $\ell_{t-1}(x)$, define $z(x) = \mathbb{I}\Big[\sum_{i=1}^G R(x, \tau_i) = 0\Big]$, i.e., $z(x) = 1$ when the group has no positive rollouts. We then increase hint strength only on such collapsed prompts:

$$\ell_t(x) = \min\{\ \ell_{t-1}(x) + 1,\ L\ \} \quad \text{if } z(x) = 1, \quad (13)$$

and otherwise keep $\ell_t(x) = \ell_{t-1}(x)$. This rule targets the specific finite-sample pathology of sparse rewards: when a group contains no positives, standardized GRPO advantages collapse and the policy-gradient estimator vanishes.

Discussion. SAGE-LIGHT (Scheme 1) is compute-efficient because it updates the hint strength only at the epoch level (no additional computation for rollouts), but it can react slowly to sudden reward collapse. SAGE (Scheme 2) is more reactive and directly targets GRPO's failure mode via a no-positives trigger, at the cost of additional probe rollouts. We report results for both schemes, and find that the no-positives trigger typically recovers faster from stalled training on hard prompts and yields better performance.

Overall, Algorithm 1 summarizes SAGE implementation. Each prompt draws *one* hint per epoch (given ℓ), and all G rollouts for that prompt share the sam context. This design reduces unnecessary variance from hint.

5. Empirical Results

Models. We use LLMs with varying degrees of math specialization: Llama-3.2-3B-Instruct (Meta, 2024), Qwen2.5-7B-Instruct (Yang et al., 2024), and Qwen3-4B-Instruct-2507 (Yang et al., 2025), representing low, moderate, and high levels of math-focused optimization, respectively, with the latter trained extensively via RL.

Training set. Our training data are drawn from OpenR1-Math-220k (Hugging Face, 2025), using prompts from NuminaMath 1.5 (Li et al., 2024) and reasoning traces generated by DeepSeek-R1 (DeepSeek-AI et al., 2025). The initial dataset contains 94k prompts. To ensure answer verifiability, we apply the *Math-Verify* tool (Kydlíček) to remove prompts whose DeepSeek-R1 reasoning traces are incorrectly verified, resulting in 64k prompts. These prompts are used in Figure 2. Due to limited resources, we further subsample 15k prompts from this set, restricting the corresponding DeepSeek-R1 reasoning traces to fewer than 8,192 tokens. This constraint is necessary because one of our baselines, LUFFY (Yan et al., 2025), relies on these reasoning traces, and excessively long traces would significantly increase RL training time. As we do not filter prompts based on pass rate, the resulting 15k prompts span a wide range of difficulty levels, resembling a practical RL training dataset.

Table 1. Accuracy on in-distribution and out-of-distribution tasks across three LLMs. The **best** and second-best results are in bold and underlined, respectively. SAGE and SAGE-LIGHT consistently outperform baselines on average across various LLMs.

Method	In-distribution							Out-of-distribution			
	AIME24 / 25	AMC23	MATH-500	Minerva	Olympiad	Avg.	Δ	GPQA	MMLU-Pro	Avg.	Δ
Llama-3.2-3B-Instruct	6.5 / 0.6	22.8	44.7	17.8	14.2	17.8	0	17.9	27.0	22.5	0
SFT	0.4 / 0.6	9.5	26.9	5.1	6.5	8.2	−9.6	11.6	18.8	15.2	−7.3
GRPO	6.7 / 0.8	29.5	52.1	20.5	21.8	21.9	+4.1	26.3	39.8	33.1	+10.6
LUFFY	4.4 / 0.4	18.6	38.9	14.3	11.9	14.7	−3.1	16.0	26.7	21.4	−1.1
Scaf-GRPO	7.7 / **2.3**	28.8	51.7	19.4	19.5	21.5	+3.7	24.1	38.0	31.0	+8.5
SAGE-LIGHT	8.8 / 1.9	32.2	54.1	**20.8**	20.1	23.0	+5.2	26.8	39.6	33.2	+10.7
SAGE	**9.2** / 0.8	**34.7**	**56.3**	20.1	**22.0**	**23.9**	**+6.1**	**27.3**	**40.7**	**34.0**	**+11.5**
Qwen2.5-7B-Instruct	13.8 / 6.7	53.4	75.7	38.1	39.2	37.8	0	37.1	56.4	46.7	0
SFT	3.5 / 7.1	30.9	56.2	20.0	21.7	23.2	−14.6	9.5	35.6	22.5	−24.2
GRPO	15.0 / **13.5**	55.5	79.2	39.1	44.5	41.1	+3.3	37.2	57.6	47.4	+0.7
LUFFY	**17.1** / **13.5**	55.2	**81.3**	39.0	44.2	41.7	+3.9	**38.1**	59.1	**48.6**	**+1.9**
Scaf-GRPO	14.6 / 12.7	58.8	78.0	**39.8**	42.0	41.0	+2.2	36.6	58.4	47.5	+0.8
SAGE-LIGHT	**17.1** / 11.7	58.1	79.9	38.6	**46.1**	41.9	+4.1	36.6	58.8	47.7	+1.0
SAGE	16.0 / 12.5	**60.3**	80.0	39.3	45.9	**42.3**	**+4.5**	38.0	**59.3**	**48.6**	**+1.9**
Qwen3-4B-Instruct	52.1 / 43.1	92.2	93.6	46.1	67.7	65.8	0	57.6	70.9	64.3	0
SFT	14.4 / 22.1	55.2	78.9	38.1	40.2	41.5	−24.3	29.4	52.6	41.0	−23.3
GRPO	55.8 / 45.0	**95.0**	96.0	**50.1**	70.4	68.7	+2.9	57.0	72.0	64.5	+0.2
LUFFY	42.3 / 36.0	86.4	91.1	48.2	59.4	60.6	−5.2	31.9	46.6	39.3	−25.0
Scaf-GRPO	**59.8** / 45.2	92.2	95.1	48.9	69.8	68.5	+2.7	54.3	72.1	63.2	−1.1
SAGE-LIGHT	59.2 / 47.1	92.2	95.1	49.5	70.5	68.9	+3.1	57.1	72.0	64.5	+0.2
SAGE	58.1 / **52.1**	94.2	**95.4**	49.2	**71.2**	**70.0**	**+4.2**	**57.8**	**72.5**	**65.2**	**+0.9**

Evaluation sets. We primarily evaluate our models on six widely used mathematical benchmarks: AIME24 (MAA Committees, 2024), AIME25 (MAA Committees, 2025), AMC23 (Li et al., 2024), MATH-500 (Hendrycks et al., 2021), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024). In addition, we include two non-mathematical benchmarks, GPQA-diamond (Rein et al., 2024) and MMLU-Pro (Wang et al., 2024), to assess the generalization ability of the trained models. **Notably, we only use hint during training. The prompt alone is the input to an LLM for evaluation.**

Baselines. We compare SAGE with the following baselines: (1) Supervised Fine-Tuning (SFT), which finetunes the model on reasoning traces from DeepSeek-R1; (2) GRPO (Shao et al., 2024), which learns without any hints; (3) LUFFY (Yan et al., 2025), which replaces one on-policy trajectory with the corresponding correct trajectory from DeepSeek-R1; and (4) Scaf-GRPO (Zhang et al., 2025b), which incorporates hints generated by GPT-5.2 under a low-reasoning-effort setting. Notably, SFT, LUFFY and Scaf-GRPO all rely on a stronger external LLM, whereas SAGE learns only from self-generated hints. For fair comparison, we reproduce LUFFY and Scaf-GRPO using their open-source implementations on the same 15k sampled prompts, aligning only the batch size and number of training steps.

Implementation details. We run all experiments on 8 A100 GPUs, and use verl (Sheng et al., 2025) for training and vLLM (Kwon et al., 2023) for sampling. Following DAPO (Yu et al., 2025), we disable the KL term by setting $\beta = 0$, and apply asymmetric clipping with $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$. Unless otherwise specified, the maximum response length is set to 8096 for both training and evaluation,[1] with a batch size of 128, 8 trajectories per prompt,[2] and 500 training steps in total. We evaluate every 50 steps, and report the best average accuracy over all checkpoints. We set L as 3, and $\alpha = 0.35$ for SAGE-LIGHT. Details of the prompt used for hint generation and injection are provided in Appendix B. Complete training and evaluation settings for all methods are reported in Appendix C.

5.1. Main results

We report results for all methods and LLMs on eight benchmarks in Table 1, with corresponding training dynamics shown in Figure 4. Across three base models, SAGE consistently achieves the highest average performance among all baselines, yielding improvements of +6.1 (Llama-3.2), +4.5 (Qwen2.5), and +4.2 (Qwen3) on average across the benchmarks. We use a fixed training set for all LLMs, despite their differing degrees of optimization for mathematical tasks. Consequently, the training set is relatively easier for Qwen3 and more challenging for Llama, a discrepancy that is also reflected in Figure 2. Nevertheless, SAGE consistently improves performance across all LLMs, demonstrating robust and effective generalization.

SAGE vs. SFT. SFT yields the worst performance, underperforming even the base LLM, due to its tendency to overfit

[1] We use 8096 for the main results, as required by LUFFY, and 2048 for the remaining experiments due to resource constraints.

[2] We use 4 trajectories for Qwen3-4B-Instruct due to slower training caused by its long response length.

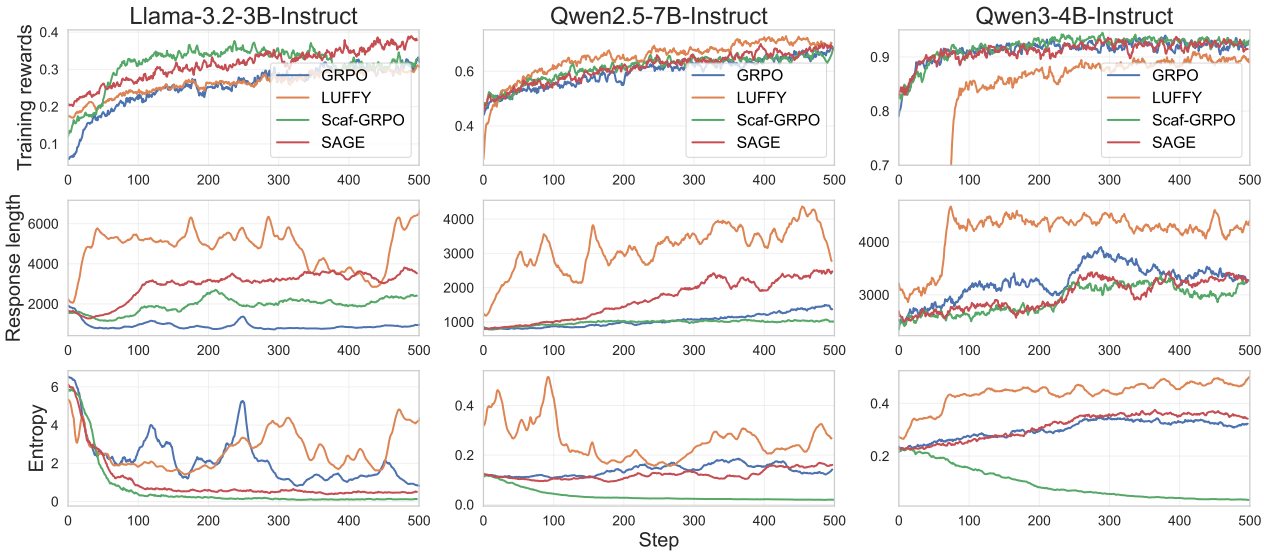

Figure 4. The training dynamics of different methods. For the training rewards, one should focus on the trend instead of the value, since adding hint (SAGE and Scaf-GRPO) modifies the prompt difficulty, and using a correct off-policy trajectory (LUFFY) increases the reward. (1) LUFFY shows the most instability, with a very high entropy for Llama and a very low reward at the beginning of training for Qwen3, because it imitates the off-policy trajectory whose distribution might not be aligned with the policy model. (2) Scaf-GRPO shows the lowest entropy, implying less exploration. (3) SAGE retains the on-policy characteristic, has a mild entropy and shows a stable growth in response length, which normally implies a better reasoning pattern.

Table 2. Percentage of prompts without any training signal, i.e., not any correct trajectories of these prompts are sampled during the whole training procedure.

Method	Llama-3.2-3B	Qwen2.5-7B	Qwen3-4B
Base	56.9%	29.8%	21.8%
GRPO	40.2%	10.3%	1.3%
SAGE	30.0%	8.2%	1.0%

training data. In contrast, SAGE preserves the RL characteristics, selectively sharpening the model's distribution to correct trajectories.

SAGE vs. GRPO. Table 2 reports the proportion of prompts that never provide a training signal. Compared to GRPO, SAGE makes substantially more effective use of the prompt set. This effect is particularly pronounced for the weaker LLM, Llama-3.2: by leveraging self-generated hints, SAGE successfully utilizes 10% more prompts, leading to the largest performance improvement over GRPO (+2.0). For the stronger model, Qwen3, SAGE behaves more similarly to GRPO, with nearly identical prompt utilization. Nevertheless, despite using only 0.3% more prompts, SAGE still achieves a +1.3 accuracy gain over GRPO. These hard prompts play a critical role in RL, which aligns with prior work (Xiong et al., 2025b; Yu et al., 2025) that favors RL on prompt sets with lower pass rates. Furthermore, in Figure 4, SAGE exhibits faster response-length growth than GRPO for both Llama-3.2 and Qwen2.5, due to learning from hard prompts that fail to provide any signal under GRPO.

SAGE vs. LUFFY. LUFFY exhibits the second-largest degree of off-policy behavior, following SFT, as one of its trajectories is generated by a different model. In Figure 4, the response length increases dramatically at the early stage of training, reflecting the LLM's tendency to imitate the stronger model. However, this off-policy setting introduces training instability due to the misalignment between the policy model and the stronger model. Specifically, Llama-3.2 trained with LUFFY displays excessively high entropy and highly oscillatory response lengths, while Qwen3 trained with LUFFY suffers from very low rewards at the beginning of training. As reported in Table 1, LUFFY only outperforms GRPO (while still underperforming SAGE) on Qwen2.5, and performs worse than the base model on both Llama-3.2 and Qwen3.

SAGE vs. Scaf-GRPO. Scaf-GRPO relies on hints generated by a stronger model (e.g., GPT-5.2 in our setting). As shown in Figure 4, it exhibits the lowest entropy among all methods, indicating limited exploration. This behavior may stem from the hints revealing excessive information. In contrast, SAGE maintains an entropy level comparable to GRPO,[3] while consistently outperforming Scaf-GRPO in Table 1. Moreover, learning from self-generated hints enables a more end-to-end training procedure and simplifies implementation.

SAGE vs, SAGE-LIGHT. SAGE-LIGHT achieves slightly lower accuracy but improves efficiency, requiring 53% of

[3]The entropy of GRPO on Llama-3.2 is abnormally high.

Table 3. Training time on Qwen2.5-7B-Instruct.

GRPO	LUFFY	Scaf-GRPO	SAGE	SAGE-LIGHT
1.0× (25.3h)	1.2×	1.5×	2.3×	1.2×

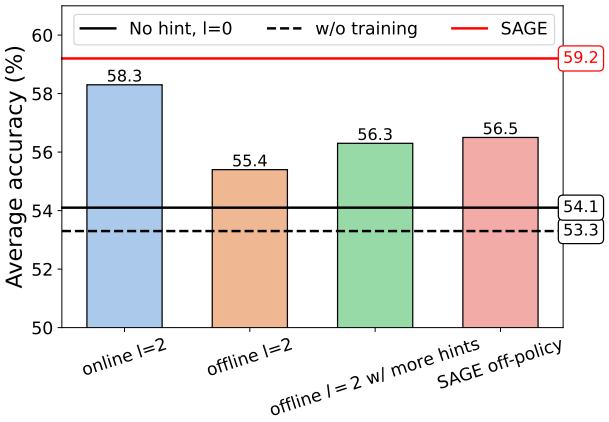

Figure 5. Ablation studies on Qwen3-4B-Instruct trained with the same prompt set as Figure 3.

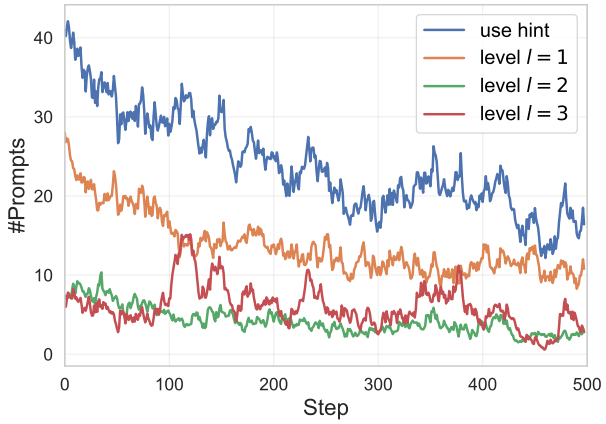

Figure 6. Number of prompts uses hint during training on Llama-3.2-3B-Instruct. Batch size is 128. The model use less hint w.r.t. the training step, indicating that the model becomes more powerful.

SAGE's training time.

Out-of-distribution performance. In Table 1, performance on out-of-distribution benchmarks shows a similar pattern to that on in-distribution ones. SAGE and SAGE-LIGHT consistently achieve the best and second-best accuracy, respectively, indicating superior generalization capability.

5.2. Discussion

Latency. A potential limitation of SAGE is its latency, as it must generate and use hints on the fly when a correct trajectory of the prompt can't be sampled. Table 3 reports the training time of different RL methods. Among them, SAGE incurs the highest training cost, while SAGE-LIGHT requires only slightly more time than GRPO. For highly complex prompts, SAGE may sample hints across multiple levels (from $l = 0$ to $l = 3$), which increases computational overhead. In contrast, SAGE-LIGHT leverages the prompt accuracy from the previous epoch to select an appropriate hint level, and thus samples from only a single level. These two SAGE variants provide flexible trade-offs for practitioners with different efficiency requirements, and both consistently outperform the baseline methods.

Offline with more hints. In Figure 3, online self-hinting generates a new hint at each training step, whereas offline self-hinting relies on a fixed hint generated prior to training. One possible explanation for the superior performance of online self-hinting is the increased diversity of hints. To examine this, we have an additional ablation (Figure 5): offline self-hinting with multiple hints. Specifically, before training, we use the base LLM to generate 10 hints with `temperature=1.0`. During training, a different hint is used for the same prompt at each step, ensuring that identical prompts are paired with diverse hints over time. We observe that increased hint diversity indeed improves performance, yielding a +0.9 gain over standard offline self-hinting. Nevertheless, online self-hinting still outperforms this variant by a margin of 2.0. We argue that hint generated in an online manner offer more benefit than diversity.

Off-policy. We ablate whether the policy-gradient matches the sampling context: SAGE optimizes $\log \pi_\theta(\cdot \mid x, h)$, while an off-policy variant rolls out with h but optimizes $\log \pi_\theta(\cdot \mid x)$. Figure 5 shows a clear drop for the off-policy variant (56.5) compared with on-policy SAGE (59.2) and even the single level hint level baseline (58.3).

Same level hint vs. SAGE. SAGE does not rely on a fixed hint level. Instead, it adaptively increases the hint level only when the current level fails to yield a correct response. In Figure 5, this strategy leads to a clear performance gain over using a constant hint level (e.g., online $l = 2$), achieving an improvement of +0.9. This design enables more effective utilization of the hard prompt, allowing the LLM to progressively learn from weaker hints, down to $l = 0$.

Less hint w.r.t. step. In Figure 6, we can observe that the LLM use less and less hint during the training. It indicates that self-hinting indeed enhances RL. LLM becomes more and more powerful and can gradually solve difficult problems without the help of a hint.

Case study. An example on how hint helps (Appendix A).

6. Related Work

Data resampling and external guidance. Data selection and filtering are widely used in online RL for LLMs (Zhang et al.; Dong et al., 2023; Xiong et al., 2023; Dong et al., 2024; Shi et al., 2024; Liao et al., 2025; Feng et al., 2025),

and become particularly important for GRPO-style methods where groupwise advantages can collapse under sparse rewards. Prior work mitigates this issue mainly by reshaping the training distribution or injecting external guidance. A common workaround is to skip degenerate groups and resample or upweight prompts (Yu et al., 2025; Xiong et al., 2025a; Yao et al., 2025; Li et al., 2025), which improves efficiency but biases training toward prompts with non-trivial success probability. Another direction bootstraps learning by adding positive trajectories from stronger teachers, reference models, or offline buffers (Zhang et al., 2025a; Yan et al., 2025), but this can introduce context or distribution mismatch when mixed with on-policy rollouts. In contrast, SAGE preserves a clean on-policy objective by changing the rollout distribution through privileged hinting, without discarding hard prompts or relying on static external buffers.

Privileged hinting and SAGE. While leveraging intermediate guidance, such as plans or gold solutions, has a rich history in RL (Ng et al., 1999; Szepesvári, 2022; Ouyang et al., 2022), recent LLM-specific adaptations often implement hinting via heuristic "batch surgery." For instance, Zhang et al. (2025b) mitigates collapse by augmenting rollout batches with hinted trajectories upon detecting failure. This approach mixes contexts (e.g., x and x, h) within a single group, which blurs the interpretation of groupwise baselines and advantage normalization. Furthermore, external hint generators may not be calibrated to the learner's current capabilities. SAGE distinguishes itself through three key design choices: (1) it formalizes hinting as an explicitly augmented on-policy sampling process, ensuring the GRPO loss remains well-defined; (2) it employs a policy-dependent strength scheduler that activates hints only when the "learning gate" is closed; and (3) it utilizes online self-hinting via a lagged policy, ensuring the hint distribution tracks the learner's evolving support rather than relying on a potentially misaligned external teacher.

7. Conclusion

We identify a finite-sample degeneracy in GRPO under sparse 0/1 rewards. When group rewards are identical, advantage standardization collapse, and the minibatch gradient vanishes on hard prompts. We propose SAGE, a privileged procedural hinting method that injects reference-solution-derived hints during training to shift the rollout distribution while preserving the original reward definition. A policy-dependent schedule gates hint strength based on detected group collapse, and inference uses the no-hint policy. Extensive experiments validate the improvements across tasks.

Acknowledgements

This research was partly supported by the Netherlands Organization for Scientific Research (NWO) under project number VI.C.192.080.

Impact Statements

This work can reduce training cost and improve stability of RL for LLMs on verifiable tasks. Risks include misuse to optimize harmful verifiable objectives.

References

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Bao, H., Xu, H., Wang, H., Ding, H., Xin, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Wang, J., Chen, J., Yuan, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Ye, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Zhao, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., SHUM, K., and Zhang, T. RAFT: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learn-*

ing Research, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=m7p5O7zblY.

Dong, H., Xiong, W., Pang, B., Wang, H., Zhao, H., Zhou, Y., Jiang, N., Sahoo, D., Xiong, C., and Zhang, T. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

Feng, Y., Kwiatkowski, A., Zheng, K., Kempe, J., and Duan, Y. Pilaf: Optimal human preference sampling for reward modeling. *arXiv preprint arXiv:2502.04270*, 2025.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z. L., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Hu, J., Wu, X., Zhu, Z., Xianyu, Wang, W., Zhang, D., and Cao, Y. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Kydlíček, H. Math-Verify: Math Verification Library. URL https://github.com/huggingface/math-verify.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Li, J., Beeching, E., Tunstall, L., Lipkin, B., Soletskyi, R., Huang, S., Rasul, K., Yu, L., Jiang, A. Q., Shen, Z., et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

Li, Z., Chen, C., Yang, T., Ding, T., Sun, R., Zhang, G., Huang, W., and Luo, Z.-Q. Knapsack rl: Unlocking exploration of llms via optimizing budget allocation. *arXiv preprint arXiv:2509.25849*, 2025.

Liao, B., Xu, Y., Dong, H., Li, J., Monz, C., Savarese, S., Sahoo, D., and Xiong, C. Reward-guided speculative decoding for efficient llm reasoning. *arXiv preprint arXiv:2501.19324*, 2025.

Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, W. S., and Lin, M. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

MAA Committees. AIME Problems and Solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions, 2024.

MAA Committees. AIME Problems and Solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions, 2025.

Meta. Introducing meta llama 3: The most capable openly available llm to date. *Meta AI Blog*, 2024. https://ai.meta.com/blog/meta-llama-3/.

Ng, A. Y., Harada, D., and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pp. 278–287. Citeseer, 1999.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y., Wu, Y., and Guo, D. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Shi, R., Zhou, R., and Du, S. S. The crucial role of samplers in online direct preference optimization. *arXiv preprint arXiv:2409.19605*, 2024.

Szepesvári, C. *Algorithms for reinforcement learning*. Springer nature, 2022.

Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

Xiong, W., Dong, H., Ye, C., Wang, Z., Zhong, H., Ji, H., Jiang, N., and Zhang, T. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. 2023.

Xiong, W., Yao, J., Xu, Y., Pang, B., Wang, L., Sahoo, D., Li, J., Jiang, N., Zhang, T., Xiong, C., et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025a.

Xiong, W., Ye, C., Liao, B., Dong, H., Xu, X., Monz, C., Bian, J., Jiang, N., and Zhang, T. Reinforce-ada: An adaptive sampling framework under non-linear rl objectives. *arXiv preprint arXiv:2510.04996*, 2025b.

Yan, J., Li, Y., Hu, Z., Wang, Z., Cui, G., Qu, X., Cheng, Y., and Zhang, Y. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Yao, J., Hao, Y., Zhang, H., Dong, H., Xiong, W., Jiang, N., and Zhang, T. Optimizing chain-of-thought reasoners via gradient variance minimization in rejection sampling and rl. *arXiv preprint arXiv:2505.02391*, 2025.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Fan, T., Liu, G., Liu, L., Liu, X., et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Zhang, C., Shen, W., Zhao, L., Zhang, X., Qi, L., Dou, W., and Bian, J. Policy filtration in rlhf to fine-tune llm for code generation.

Zhang, K., Lv, A., Li, J., Wang, Y., Wang, F., Hu, H., and Yan, R. Stephint: Multi-level stepwise hints enhance reinforcement learning to reason. *arXiv preprint arXiv:2507.02841*, 2025a.

Zhang, X., Wu, S., Zhu, Y., Tan, H., Yu, S., He, Z., and Jia, J. Scaf-grpo: Scaffolded group relative policy optimization for enhancing llm reasoning. *arXiv preprint arXiv:2510.19807*, 2025b.

Zhang, Y., Yao, W., Yu, C., Liu, Y., Yin, Q., Yin, B., Yun, H., and Li, L. Improving sampling efficiency in rlvr through adaptive rollout and response reuse. *arXiv preprint arXiv:2509.25808*, 2025c.

A. Illustration of Privileged Hinting

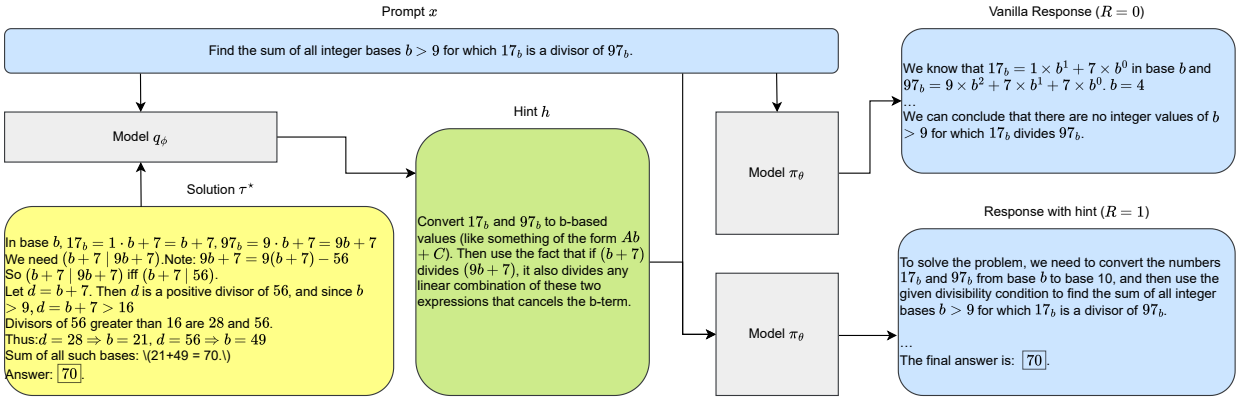

Figure A.1. Example of privileged hinting for a single prompt. Given a math prompt x, a hint generator q_ϕ uses the reference solution $\tau^\star$ during training to produce a procedural hint (h) that summarizes intermediate reasoning without revealing the final answer. Rolling out the policy π_θ on the original prompt can yield an incorrect solution with zero reward, while conditioning on (x, h) shifts the rollout distribution and enables a correct solution with positive reward. The task return is unchanged, and at deployment the hint is removed so the model runs on the original prompt only.

Case Study: Progressive privileged hints for $17_b \mid 97_b$

Prompt. Find the sum of all integer bases $b > 9$ for which 17_b is a divisor of 97_b.

Reference solution (available only during training). In base b,

$$17_b = 1 \cdot b + 7 = b + 7, \qquad 97_b = 9 \cdot b + 7 = 9b + 7.$$

We need $b + 7 \mid 9b + 7$. Note that

$$9b + 7 = 9(b + 7) - 56,$$

hence $b + 7 \mid 9b + 7$ iff $b + 7 \mid 56$. Let $d = b + 7$. Then d is a positive divisor of 56, and since $b > 9$ we have $d = b + 7 > 16$. The divisors of 56 greater than 16 are 28 and 56, so $b \in \{21, 49\}$ and the sum is 70.

Privileged hints (used only during training; never shown at test time).

- **Level 1 (minimal).** Rewrite the base-b numerals as ordinary integers in terms of b, then turn the divisibility condition into a statement about a simple linear expression.
- **Level 2 (medium).** Convert 17_b and 97_b into the form $Ab + C$. If $(b + 7)$ divides $(9b + 7)$, it also divides any linear combination of these expressions that cancels the b-term.
- **Level 3 (detailed).** Compute $17_b = b + 7$ and $97_b = 9b + 7$. Subtract a multiple of $(b + 7)$ from $(9b + 7)$ to eliminate b, e.g., $(9b + 7) - 9(b + 7)$. The condition becomes "$(b + 7)$ divides a constant". Enumerate divisors and keep only $b > 9$.

Representative model behaviors (illustrative).

- **No hint ($h = \varnothing$):** the model may mis-expand 97_b (e.g., treating it as a multi-digit polynomial), and incorrectly conclude there is no valid $b > 9$, yielding a wrong terminal decision and thus reward 0.
- **With stronger hints (Level 2/3):** the model is steered toward the key cancellation step $9b + 7 - 9(b + 7) = -56$, after which it can enumerate divisors and recover the correct bases and final sum (70).

Case Study: Privileged Hinting on a Simple Verifiable Math Task. To make the idea of *privileged hinting* concrete, we include a small case study on a verifiable divisibility question. The key point is that hints are *training-time privileged context*: they do not modify the verifier or the terminal reward. Instead, they reshape the rollout distribution so that, under finite sampling, the policy is more likely to generate informative trajectories (e.g., those that perform the correct algebraic reduction). In practice, without hints the model may repeatedly take an incorrect representation path (e.g., mis-expanding base-b numerals) and receive identical zero rewards across a rollout group, causing GRPO advantages to collapse. With

progressive hints, the model is guided toward the correct reduction, increasing the chance that a group contains mixed outcomes and thus yields a non-degenerate update.

This example matches the operational role of SAGE: hints do not change the verifier reward, but they increase the probability that at least one rollout in a finite group follows a *useful* trajectory (here, the cancellation-and-divisor-enumeration path). This increases within-group outcome diversity, reduces the frequency of degenerate all-zero groups, and therefore prevents standardized GRPO advantages from collapsing on hard prompts under finite sampling.

B. Prompt for hint generation and the usage of hint

System prompt for hint generation

You are a tutoring assistant that generates progressive hints to help students solve difficult problems without revealing the solution directly.

TASK:
Given a question and its solution, generate 3 levels of hints that progressively guide the student toward solving the problem independently.

HINT LEVELS:
- Level 1: Minimal hint - Points to the key concept or approach without specifics
- Level 2: Medium hint - Provides more direction on the method or intermediate steps
- Level 3: Detailed hint - Gives substantial guidance while still requiring the student to complete the solution

GUIDELINES:
- Never reveal the final answer
- Each level should be built on the previous one
- Hints should inspire problem-solving, not just provide steps to copy
- Tailor hint difficulty to bridge the gap between the student's level and the solution

OUTPUT FORMAT:

```
```json
{
  "level_1": "minimal hint text",
  "level_2": "medium hint text",
  "level_3": "detailed hint text"
}
```
```

User prompt for hint generation

Question:
{problem}

Solution:
{solution}

System prompt for RL

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

User prompt for RL

{problem}

Here is a hint to help you:
{hint}

C. Detailed implementation settings

C.1. Training settings

SFT. We use OpenRLHF (Hu et al., 2024) for SFT, and set the learning rate as 5e-5, the batch size as 64, warmup ratio as 10%, and number of epochs as 3. We evaluate on the final checkpoints.

GRPO. GRPO shares the same training settings as SAGE, as stated in §5.

LUFFY. We use the open-source implementation[4] to reproduce LUFFY, and set the batch size as 128, and *ppo_mini_batch_size* as 64. These two hyper-parameters stay the same for all RL methods.

Scaf-GRPO. We use the open-source implementation[5] to reproduce Scaf-GRPO.

C.2. Evaluation settings

For the main results in Table 1, we evaluate all models with a max response length of 8192, *temperature=0.6* and *top_p=0.95*. For the rest, we set a max response length of 2048.

Table C.1. Detailed number for Figure 3 (Left). By default, we train for 200 steps, but for 400 steps for methods denoted by *. The results for level $l = 2$ in Figure 3 are from methods denoted by *. Different from the default training setting of SAGE, we set *ppo_mini_batch_size=32* here.

Method	Hint level	AIME24	AIME25	AMC23	MATH-500	Minerva	Olympiad	Avg.
Qwen3-4B-Instruct	-	30.8	28.5	75.5	87.8	42.8	54.3	53.3
No hint	0	34.0	29.2	78.8	88.6	40.7	53.1	54.1
GPT-5 hint	1	34.0	29.8	83.0	89.6	42.2	56.9	55.9
Self-hint offline	1	35.8	28.3	80.2	89.9	42.4	56.9	55.6
Self-hint online	1	35.0	28.5	82.7	90.0	42.7	61.0	56.7
GPT-5 hint	2	34.8	29.6	81.1	89.6	42.9	60.9	56.5
Self-hint offline	2	33.1	27.3	78.1	89.0	42.5	59.2	54.9
Self-hint online	2	34.2	27.7	81.7	89.2	43.5	61.8	56.4
GPT-5 hint*	2	34.8	29.6	81.1	89.6	42.9	60.9	56.5
Self-hint offline*	2	34.0	26.0	79.8	88.5	43.7	60.4	55.4
Self-hint online*	2	38.5	30.8	82.7	90.3	44.4	62.9	58.3
GPT-5 hint	3	37.9	26.9	78.3	89.2	43.1	62.1	56.3
Self-hint offline	3	36.7	27.5	83.1	89.6	43.3	61.5	57.0
Self-hint online	3	36.7	27.2	83.1	89.9	43.2	62.6	57.1

D. Proofs

This appendix provides detailed proofs for the results in Section 3.1 (and additional analysis).

[4]https://github.com/ElliottYan/LUFFY
[5]https://github.com/JIA-Lab-research/Scaf-GRPO

D.1. Preliminaries: Bernoulli groups and sample variance

Fix a context (x, h) and draw $G \geq 2$ i.i.d. rollouts with terminal rewards $R_i \in \{0, 1\}$. Let

$$\bar{R} \coloneqq \frac{1}{G}\sum_{i=1}^{G} R_i, \qquad s^2 \coloneqq \frac{1}{G}\sum_{i=1}^{G}(R_i - \bar{R})^2, \qquad s \coloneqq \sqrt{s^2}.$$

When standardized GRPO is used, advantages are

$$A_i \coloneqq \frac{R_i - \bar{R}}{s + \epsilon},$$

where $\epsilon > 0$ is a numerical stabilizer (as used in the main text).

We will repeatedly use the fact that since $R_i \in \{0, 1\}$,

$$\sum_{i=1}^{G}(R_i - \bar{R})^2 = \sum_{i=1}^{G} R_i^2 - G\bar{R}^2 = \sum_{i=1}^{G} R_i - G\bar{R}^2 = G\bar{R} - G\bar{R}^2 = G\bar{R}(1 - \bar{R}), \tag{14}$$

hence

$$s^2 = \bar{R}(1 - \bar{R}). \tag{15}$$

In particular, $s^2 = 0$ iff $\bar{R} \in \{0, 1\}$, i.e., iff all R_i are identical.

D.2. Proof of Corollary 3.1

Proof. Recall the advantage energy

$$E \coloneqq \frac{1}{G}\sum_{i=1}^{G} A_i^2, \qquad A_i = \frac{R_i - \bar{R}}{s + \epsilon}.$$

For $\epsilon > 0$, we compute

$$E = \frac{1}{G}\sum_{i=1}^{G}\frac{(R_i - \bar{R})^2}{(s+\epsilon)^2} = \frac{1}{(s+\epsilon)^2} \cdot \frac{1}{G}\sum_{i=1}^{G}(R_i - \bar{R})^2 = \frac{s^2}{(s+\epsilon)^2},$$

which is exactly Eq. (6).

Since $s \geq 0$ and $\epsilon > 0$, we have $0 \leq \frac{s}{s+\epsilon} < 1$, hence

$$0 \leq E = \left(\frac{s}{s+\epsilon}\right)^2 < 1,$$

so $E \in [0, 1)$ (and in the limit $\epsilon \to 0^+$, $E \to \mathbb{I}[s > 0]$).

Monotonicity in s follows by differentiation: define $f(s) \coloneqq \frac{s^2}{(s+\epsilon)^2}$ for $s \geq 0$. Then

$$f'(s) = \frac{2s(s+\epsilon)^2 - s^2 \cdot 2(s+\epsilon)}{(s+\epsilon)^4} = \frac{2s\epsilon}{(s+\epsilon)^3} \geq 0,$$

so E is non-decreasing in s.

Finally, if the group is degenerate, then $s = 0$ and therefore $E = 0$. This shows the standardized signal energy collapses to 0 exactly when within-group variance collapses. □

D.3. Proof of Proposition 3.2

Proof. Write $p \coloneqq p_\theta(x, h) = \Pr[R(x, \tau) = 1 \mid x, h]$. Then $R_1, \ldots, R_G$ are i.i.d. Bernoulli(p).

As noted in Section D.1, $s = 0$ iff all rewards are identical. There are exactly two degenerate cases: (i) all-zero: $R_1 = \cdots = R_G = 0$; (ii) all-one: $R_1 = \cdots = R_G = 1$. Thus

$$\Pr[s > 0 \mid x, h] = 1 - \Pr[\text{all-zero}] - \Pr[\text{all-one}] = 1 - (1-p)^G - p^G,$$

which proves Eq. (7).

To locate the maximizer, define $u(p) \coloneqq 1 - (1-p)^G - p^G$ on $[0, 1]$. We have symmetry $u(p) = u(1-p)$. Moreover, for $G \geq 2$,

$$u''(p) = -\,G(G-1)\Big(p^{G-2} + (1-p)^{G-2}\Big) < 0,$$

so u is strictly concave, hence has a unique maximizer. By symmetry, the unique maximizer must be at $p = \frac{1}{2}$.

Finally, in the sparse regime $p \ll 1$,

$$u(p) = 1 - (1-p)^G - p^G = 1 - \Big(1 - Gp + O(p^2)\Big) - O(p^G) = Gp + O(p^2),$$

so $\Pr[s > 0 \mid x, h] \approx Gp$. □

D.4. Proof of Proposition 3.3

Proof. Fix x and $G \geq 2$. Recall $u(p) = 1 - (1-p)^G - p^G$ and

$$J_x(\theta, q) = \mathbb{E}_{h \sim q(\cdot|x)}\Big[u\big(p_\theta(x, h)\big)\Big].$$

We first verify the claims about u. Symmetry: $u(1-p) = 1 - p^G - (1-p)^G = u(p)$. For strict concavity, compute for $G \geq 2$:

$$u''(p) = -\,G(G-1)\Big(p^{G-2} + (1-p)^{G-2}\Big) < 0,$$

so u is strictly concave on $[0, 1]$. By symmetry and strict concavity, the unique maximizer is $p = \frac{1}{2}$.

For a fixed θ, define the measurable function

$$v_\theta(h) \coloneqq u(p_\theta(x, h)) \in [0, 1].$$

Then $J_x(\theta, q) = \mathbb{E}_{h \sim q}[v_\theta(h)]$. Over all probability distributions $q(\cdot \mid x)$ supported on the admissible hint space, the maximizers must concentrate probability mass on (essential) maximizers of v_θ: indeed, since the objective is linear in q, any optimizer can be chosen to put all mass on

$$\arg\max_h v_\theta(h) = \arg\max_h u(p_\theta(x, h))\,.$$

Because u is uniquely maximized at $p = \frac{1}{2}$ and is strictly decreasing as p moves away from $\frac{1}{2}$ (due to strict concavity and symmetry), the maximizers of $v_\theta(h)$ are exactly the *calibrating hints* that make $p_\theta(x, h)$ as close as possible to $\frac{1}{2}$ (and equal to $\frac{1}{2}$ when achievable). This proves the statement that an optimal $q^\star_\theta(\cdot \mid x)$ places its mass on calibrating hints.

In general, $p_\theta(x, h)$ changes with θ because the rollout distribution under $\pi_\theta(\cdot \mid x, h)$ changes with θ. Therefore the set of calibrating hints

$$\mathcal{H}^\star_\theta(x) \coloneqq \arg\max_h u(p_\theta(x, h))$$

typically varies with θ. Unless $p_\theta(x, h)$ (hence $v_\theta(h)$) is invariant in θ for q-almost all h, a fixed distribution q that was optimal (or near-optimal) early in training will drift away from being optimal later, reducing $J_x(\theta, q)$ relative to a θ-adapted choice. This formalizes why updating the hint generator online using the policy can reduce gate-mismatch. □

D.5. Proof of the Jensen inequality in Remark 1

Proof. Fix $G \geq 2$ and define $u(p) = 1 - (1-p)^G - p^G$. We showed above that u is concave on $[0,1]$ because $u''(p) \leq 0$. Let $Z \coloneqq p_\theta(x,h) \in [0,1]$ be the random success probability induced by sampling $h \sim q$. Then Jensen's inequality for concave u gives

$$\mathbb{E}_h\big[u(Z)\big] \leq u(\mathbb{E}_h[Z])\,,$$

which is exactly Eq. (11). Equality holds only when Z is almost surely constant (or when u is affine on the support, which does not happen for $G \geq 2$ except in degenerate cases). This shows that, at a fixed mean success probability, additional variability across hints can only decrease the expected gate-opening frequency. □

D.6. A sharper small-p expansion of the gate probability

For completeness, we also record an exact decomposition that makes the Gp scaling explicit. Let $p = \Pr[R = 1 \mid x, h]$. Then

$$\begin{aligned}\Pr[s > 0 \mid x, h] &= 1 - (1-p)^G - p^G \\ &= \sum_{k=1}^{G-1} \binom{G}{k} p^k (1-p)^{G-k}. \end{aligned} \tag{16}$$

When $p \ll 1$, the dominant term is $k = 1$:

$$\Pr[s > 0 \mid x, h] = Gp(1-p)^{G-1} + O(p^2) \approx Gp,$$

and the neglected p^G term is exponentially smaller in G.

D.7. Non-standardized GRPO signal energy

Non-standardized advantage is also widely used in GRPO-like algorithms (Liu et al., 2025), which also provide insights about the behavior.

Setup. Define non-standardized (mean-centered) advantages

$$\tilde{A}_i \coloneqq R_i - \bar{R}, \qquad \bar{R} = \frac{1}{G}\sum_{i=1}^{G} R_i,$$

and define the (non-standardized) energy

$$\tilde{E} \coloneqq \frac{1}{G}\sum_{i=1}^{G} \tilde{A}_i^2.$$

Note that $\tilde{E} = s^2$ by definition.

Proposition D.1 (Expected non-standardized energy under Bernoulli rewards)**.** *Conditioned on (x,h) with success probability $p = p_\theta(x,h)$,*

$$\mathbb{E}\big[\tilde{E} \mid x, h\big] = \frac{G-1}{G}\, p(1-p). \tag{17}$$

Proof. Let $S = \sum_{i=1}^{G} R_i$ so that $\bar{R} = S/G$. Using identity (14),

$$\tilde{E} = \frac{1}{G}\sum_{i=1}^{G}(R_i - \bar{R})^2 = \bar{R}(1-\bar{R}).$$

Taking expectation:

$$\mathbb{E}[\tilde{E}] = \mathbb{E}[\bar{R}] - \mathbb{E}[\bar{R}^2].$$

We have $\mathbb{E}[\bar{R}] = p$. Also,

$$\mathbb{E}[\bar{R}^2] = \mathrm{Var}(\bar{R}) + (\mathbb{E}[\bar{R}])^2 = \frac{1}{G^2}\mathrm{Var}(S) + p^2 = \frac{1}{G^2}\cdot Gp(1-p) + p^2 = \frac{p(1-p)}{G} + p^2.$$

Therefore

$$\mathbb{E}[\tilde{E}] = p - \left(\frac{p(1-p)}{G} + p^2\right) = \frac{G-1}{G}\,p(1-p).$$

□

Proposition D.2 (Optimal calibrated difficulty for mean-centered updates)**.** *For fixed $G \geq 2$, the right-hand side of Eq.* (17) *is uniquely maximized at $p = \frac{1}{2}$.*

Proof. Let $f(p) = p(1-p) = p - p^2$. Then $f'(p) = 1 - 2p$ and $f''(p) = -2 < 0$, so f is strictly concave and uniquely maximized at $p = \frac{1}{2}$. The prefactor $(G-1)/G$ does not affect the maximizer. □

Remark D.3 (Hint randomness incurs a variance penalty at fixed mean). Let $Z = p_\theta(x, h)$ be random due to $h \sim q$ and $\bar{p} = \mathbb{E}[Z]$. Then

$$\mathbb{E}\big[Z(1-Z)\big] = \bar{p}(1-\bar{p}) - \mathrm{Var}(Z).$$

Thus, at fixed mean success rate, additional variability in $p_\theta(x, h)$ across hints reduces the expected energy.

Proof. Compute

$$\mathbb{E}[Z(1-Z)] = \mathbb{E}[Z] - \mathbb{E}[Z^2] = \bar{p} - \big(\mathrm{Var}(Z) + \bar{p}^2\big) = \bar{p}(1-\bar{p}) - \mathrm{Var}(Z).$$

□