EXPANDING THE CAPABILITIES OF REINFORCEMENT LEARNING VIA TEXT FEEDBACK

Yuda Song[*,1] **Lili Chen**[*,1] **Fahim Tajwar**[1] **Rémi Munos**[2] **Deepak Pathak**[1]
J. Andrew Bagnell[1,3] **Aarti Singh**[1] **Andrea Zanette**[1]
[1]Carnegie Mellon University [2]Inria [3]Aurora Innovation [*]Equal contribution
{yudas,lilic}@andrew.cmu.edu

ABSTRACT

The success of RL for LLM post-training stems from an unreasonably uninformative source: a single bit of information per rollout as binary reward or preference label. At the other extreme, distillation offers dense supervision but requires demonstrations, which are costly and difficult to scale. We study text feedback as an intermediate signal: richer than scalar rewards, yet cheaper than complete demonstrations. Textual feedback is a natural mode of human interaction and is already abundant in many real-world settings, where users, annotators, and automated judges routinely critique LLM outputs. Towards leveraging text feedback at scale, we formalize a multi-turn RL setup, RL from Text Feedback (RLTF), where text feedback is *available during training but not at inference*. Therefore, models must learn to internalize the feedback in order to improve their test-time single-turn performance. To do this, we propose two methods: **Self Distillation** (RLTF-SD), which trains the single-turn policy to match its own feedback-conditioned second-turn generations; and **Feedback Modeling** (RLTF-FM), which predicts the feedback as an auxiliary objective. We provide theoretical analysis on both methods, and empirically evaluate on reasoning puzzles, competition math, and creative writing tasks. Our results show that both methods consistently outperform strong baselines across benchmarks, highlighting the potential of RL with an additional source of rich supervision at scale.

1 INTRODUCTION

Reinforcement learning (RL) has become the foundational technique in modern LLM post-training, often delivering large gains in instruction-following, helpfulness, and reasoning quality (Ouyang et al., 2022; Guo et al., 2025a). Yet the standard RL signal in these systems is typically a sparse scalar reward (or one-bit preference label) per rollout. This creates a fundamental tension: RL can be remarkably effective at scale, but the outcome of each individual trajectory contains very little information about what went wrong and how to fix it, making learning extremely inefficient when the base model is unable to solve the task.

At the other extreme, distillation (Buciluǎ et al., 2006; Ba & Caruana, 2014; Hinton et al., 2015) and imitation learning provide information-dense supervision: a single demonstration can convey a full solution, or token-level correction for on-policy imitation learning (Ross et al., 2011; Agarwal et al., 2024; Lu & Lab, 2025). However, distillation is not applicable to training frontier models, and collecting demonstrations from humans is not scalable.

Natural-language text feedback is both a natural mode of human interaction and already abundant in practice. Users routinely critique chatbot outputs; tool-mediated workflows (e.g., code execution, unit tests, compiler errors) produce structured traces that describe failures; and more broadly, natural-language feedback is the primary medium through which humans teach and correct one another. Beyond its abundance, text feedback also occupies a favorable middle ground in information density, offering the best of both worlds: it is richer than a scalar reward, yet cheaper than a complete demonstration—feedback can localize an error, name a violated constraint, or suggest a fix.

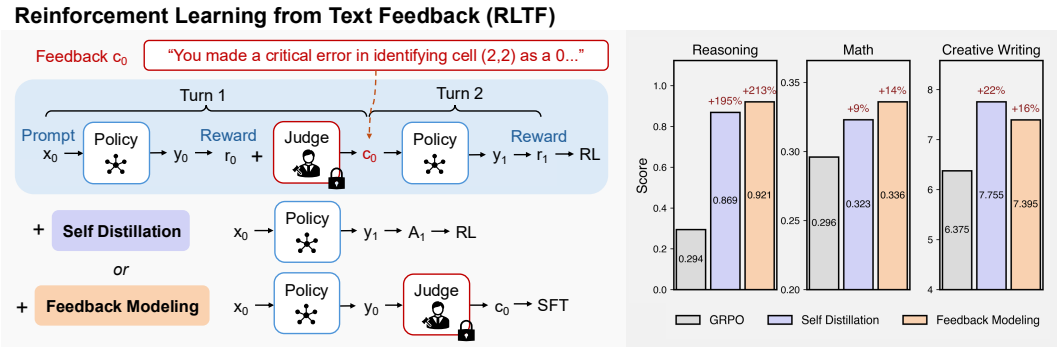

Figure 1: Left: performance of our two methods, Self Distillation and Feedback Modeling, on **reasoning puzzles**, **competition math**, and **creative writing**. Both methods outperform standard single-turn GRPO. Right: overview of RL from text feedback, which uses a feedback provider (judge) to generate critiques c_0. Self Distillation trains the policy to match the feedback-conditioned second-turn generations y_1, and Feedback Modeling predicts the critiques c_0 as an auxiliary objective.

A natural framework for incorporating feedback is multi-turn interaction: the model generates an attempt, feedback is appended to form an extended prompt, and the model revises. One might apply standard multi-turn RL to this setting, treating the conversation as a sequential decision-making problem and optimizing cumulative reward across turns (Zhou et al., 2024; Shani et al., 2024). However, this creates a fundamental asymmetry: during training, feedback can guide revision, but at test time, feedback is often unavailable—users want good outputs on the first try, not a back-and-forth dialogue. Without feedback, the second turn is not even well-defined.[1] With naive multi-turn RL, the policy learns to respond well—leveraging it for *test-time refinement with feedback*—but this does not translate into better first-turn performance when feedback is unavailable. Empirically, we find that naive multi-turn RL improves second-turn performance but yields little gain on the first turn (cf. Section 5). To *internalize* feedback rather than merely condition on it, we need learning objectives that explicitly transform feedback into first-turn supervision. By internalizing text feedback during training, RL can succeed in settings where sparse scalar rewards alone provide insufficient learning signal—effectively expanding the frontier of what reinforcement learning can accomplish.

To address this challenge, we study the setting of RL from Text Feedback (RLTF) and propose two methods to improve first-turn performance from training-time feedback: Self Distillation (RLTF-SD), which treats feedback-conditioned second attempts as implicit demonstrations for the single-turn policy, and Feedback Modeling (RLTF-FM), which learns from feedback itself by predicting critiques as an auxiliary objective. Notably, RLTF-FM can elicit test-time refinement *without* feedback, by rolling out the model's self-critiques during inference. Concretely, our contributions are as follows:

- A formalization of reinforcement learning from text feedback for improving single-turn test-time performance by using feedback during training.
- Two methods to incorporate text feedback into model capabilities: RLTF-SD and RLTF-FM.
- A theoretical justification of our design choices for RLTF-SD, and an extensive theoretical analysis of RLTF-FM via the lens of representation learning.
- Empirical investigation with extensive comparisons and ablations on a suite of diverse benchmarks: Reasoning Gym (Stojanovski et al., 2025), MATH500 (Hendrycks et al., 2021), AIME24, LitBench (Fein et al., 2025) and WritingBench (Wu et al., 2025). Our experiments demonstrate that both of our proposed methods significantly improve single-turn test-time performance over strong baselines that use rewards and text feedback.

[1]This highlights a significant distinction between different modes of text feedback: while some text feedback such as code execution is available during test time (e.g., through tool use), most text feedback such as human feedback is unavailable during test time. In this work we focus on the latter and more challenging setting, and study how to generalize when the feedback is unavailable during test time.

2 RL FROM TEXT FEEDBACK

Let $\mathcal{X}$ be the prompt space and let $\mathcal{X}_0 \subset \mathcal{X}$ be the set of initial prompts that defines the task. Let $\mu \in \Delta(\mathcal{X})$ be the distribution of the initial prompts. We use $\mathcal{Y}$ to denote the output space, and an (LLM) policy π maps prompts to distributions over outputs, i.e., $\pi : \mathcal{X} \to \Delta(\mathcal{Y})$. Similarly, let $\mathcal{C}$ be the text feedback space and $\mathcal{M}$ a text feedback provider (human, interpreter, etc.). $\mathcal{M}$ samples text feedback given a prompt and output, i.e., $\mathcal{M} : \mathcal{X} \times \mathcal{Y} \to \Delta(\mathcal{C})$. Finally, let $R : \mathcal{X}_0 \times \mathcal{Y} \to [0, 1]$ be the reward function (that is always evaluated on the original prompt) and H be the horizon.

Interaction protocol. At the first timestep $h = 0$, a prompt $x_0 \sim \mu(\mathcal{X}_0)$ is sampled, the policy samples an output $y_0 \sim \pi(\cdot \mid x_0)$, receives reward $r_0 = R(x_0, y_0)$, and the feedback provider supplies $c_0 \sim \mathcal{M}(\cdot \mid x_0, y_0)$. For any timestep $h > 0$, the prompt is updated as a function of previous information, $x_h = f(x_{h-1}, y_{h-1}, c_{h-1})$, where f may be as simple as concatenation (e.g., appending y_{h-1} and c_{h-1} to the conversation). The rest follows the first turn: the policy samples $y_h \sim \pi(\cdot \mid x_h)$, obtains reward $r_h = R(x_0, y_h)$, and receives feedback $c_h \sim \mathcal{M}(\cdot \mid x_h, y_h)$. In realistic deployments, $\mathcal{M}$ (or the environment) may terminate the episode early when $r_h = 1$; we will make this explicit whenever early termination is used. We use $\mathbb{P}^\pi$ and $\mathbb{E}^\pi$ to denote the law and expectation over the above interaction process induced by π, $\mathcal{M}$, and f.

Learning objective. One natural objective is to maximize the expected sum of rewards over the H-turn interaction:

$$J_{\mathsf{MultiTurn}}(\pi) = \mathbb{E}^\pi \left[\sum_{h=0}^{H-1} r_h \right]. \tag{1}$$

This can be optimized using standard multi-turn RL algorithms by treating the interaction as an episodic MDP over the augmented prompts x_h (Zhou et al., 2024; Shani et al., 2024). However, it alone does not isolate the role of text feedback: the policy could maximize reward while treating c_h merely as additional context. Indeed, the objective remains well-defined even if feedback were replaced by uninformative tokens; the policy might simply learn to ignore it. We verify this empirically in Section 5, where naive RL improves multi-turn performance but yields little gain on the first turn.

RL from text feedback (RLTF). Our goal is to leverage text feedback as a *learning* signal that improves the model's single-turn competence, not merely its ability to improve *test-time refinement with feedback*. To formalize this, we define the single-turn objective:

$$J_{\mathsf{SingleTurn}}(\pi) = \mathbb{E}_{x_0 \sim \mu} \left[\mathbb{E}_{y \sim \pi(\cdot \mid x_0)} [R(x_0, y)] \right], \tag{2}$$

which evaluates the policy on initial prompts x_0 without additional feedback at test time. In RLTF, while optimizing Eq. (1) is straightforward with multi-turn RL algorithms, the central research question is then: *Given access to feedback-augmented trajectories τ during training, how can we design learning objectives and algorithms that improve $J_{\mathsf{SingleTurn}}(\pi)$?*

We address this question with two complementary methods, described in Sections 3 and 4.

3 SELF DISTILLATION

Text feedback is particularly valuable because it often turns an incorrect first attempt into a correct second attempt: after receiving a critique, the same policy can revise its answer and improve. Our goal is to convert this *feedback-conditioned* competence into improvement on the *single-turn* metric (Eq. (2)), so that the policy performs well even when feedback is unavailable at test time. We propose to do this via Self Distillation: we treat the policy acting under the post-feedback prompt as an implicit teacher, and distill it into the original one-shot policy. In this sense, distillation "compiles away" the need for feedback by turning test-time refinement into a training signal. This gives us higher-quality trajectories than sampling directly from $\pi(\cdot \mid x_0)$, reducing the exploration burden and turning sparse reward learning into learning from corrected solutions.

Concretely, (focusing on the two-turn case,) for each initial prompt x_0 we sample a first-turn output $y_0 \sim \pi(\cdot \mid x_0)$, obtain feedback c_0, and form the feedback-augmented prompt $x_1 = f(x_0, y_0, c_0)$. We then sample a revised output $y_1 \sim \pi(\cdot \mid x_1)$ and use y_1 to update $\pi(\cdot \mid x_0)$ (not $\pi(\cdot \mid x_1)$), thereby directly targeting single-turn performance. This leads to the following RL-style distillation objective

that learns from the y_1 distribution:

$$\ell_{\mathsf{distill}}(\pi) = \mathbb{E}_{x_1 \sim \mathbb{P}^\pi, y_1 \sim \pi(\cdot|x_1)} \left[\frac{\pi(y_1 \mid x_0)}{\pi_{\text{ref}}(y_1 \mid x_1)} A(x_0, y_1) \right]. \tag{3}$$

Here π_{ref} denotes a reference distribution used for correction, and $A(x_0, y_1)$ is an estimator of the reward $R(x_0, y_1)$[2], and sg denotes the stop-gradient operator. In the following we will omit the dependency on $x_1 \sim \mathbb{P}^\pi$ when it is clear from context. We introduce Eq. (3) to unify several natural objectives with different instantiations of π_{ref} and $A(\cdot)$.

When we set $\pi_{\text{ref}}(\cdot \mid x_1) = \pi(\cdot \mid x_1)$ (i.e., the data-collection distribution for y_1), Eq. (3) recovers an off-policy objective with importance-sampling correction. Moreover, taking $A(y_1) = R(x_0, y_1)$ recovers the original single-turn objective in expectation:

$$\mathbb{E}_{y_1 \sim \pi(\cdot|x_1)} \left[\frac{\pi(y_1 \mid x_0)}{\pi(y_1 \mid x_1)} R(x_0, y_1) \right] = J_{\mathsf{SingleTurn}}(\pi). \tag{4}$$

Taking gradient with respect to π on Eq. (4) gives an unbiased gradient estimator for $\nabla J_{\mathsf{SingleTurn}}(\pi)$ (under the standard support condition, and note that this in general does not hold for other choices of π_{ref}). Thus, we can obtain an (unbiased) gradient for the *single-turn* objective $J_{\mathsf{SingleTurn}}(\pi)$ using samples from the *second-turn* policy, effectively leveraging feedback-conditioned rollouts to improve first-attempt performance. All derivations and proofs from this section can be found in Section B.

Baselines. For stability and efficiency of policy gradient algorithms, baseline design (i.e., control variates) is crucial to reduce the variance of the policy gradient estimator (Williams, 1992; Schulman et al., 2015; Guo et al., 2025a; Zeng et al., 2025). Thus it is important to derive the most effective baselines for the distillation objective (Eq. (3)) as it inherits the form of the policy gradient objective. A natural choice is to use the GRPO-style (Guo et al., 2025a) group-mean baseline computed from second-turn rewards, as this is the standard baseline in multi-turn LLM RL (Team et al., 2025; Tan et al., 2025). However, this baseline suffers from an in-sample coupling issue and gradient-signal collapse. A second-turn group-mean baseline centers rewards using the same second-turn samples: this can be unbiased in expectation, but it exhibits a *point-wise* degeneracy: whenever the group rewards are (nearly) constant, the centered reward-estimations vanish and the update is exactly (or approximately) zero. We discuss this in more detail in Section B.

First-turn baseline. Baselines computed based on first-turn quantities do not suffer from the above in-sample coupling with y_1^i, or the gradient collapse issue. Let

$$b^{(0)} := \frac{1}{N} \sum_{j=1}^{N} R(x_0, y_0^j), \quad A_i^{(0)} := R(x_0, y_1^i) - b^{(0)}, \tag{5}$$

we have (with $\pi_{\text{ref}}(\cdot) = \pi(\cdot \mid x_1)$) that the baseline term $b^{(0)}$ is 0 in expectation, and therefore

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)} A_i^{(0)} \nabla \log \pi(y_1^i \mid x_0) \right] = \nabla J_{\mathsf{SingleTurn}}(\pi).$$

In addition, the first-turn baseline $b^{(0)}$ does not normalize by post-feedback rewards. When p_1 is high but the first-turn policy is still imperfect ($b^{(0)} < 1$), we have $A_i^{(0)} = R(x_0, y_1^i) - b^{(0)} \neq 0$, so the update remains non-trivial and only vanishes when the student itself is already correct. Note that another natural variant that avoids this issue is taking $A_i = R(x_0, y_1^i) - R(x_0, y_0^i)$, which can be interpreted as the improvement from feedback for each specific trajectory, but potentially result in higher variance than Eq. (5). We defer detailed discussions to Sections B.2 and B.3.

Bias-variance tradeoff in importance weighting. Recall that an unbiased estimator of the single-turn policy gradient can be obtained by importance weighting with $\pi_{\text{ref}} = \pi(\cdot \mid x_1)$, but its variance is controlled by the second moment, an expectation over the directly policy ratios between $\pi(\cdot \mid x_1)$ and $\pi(\cdot \mid x_0)$, instead of the logarithmic of the ratio. Therefore, even moderate distribution shift between first- and second-turn policies can induce heavy-tailed weights. For LLM outputs y (long token

[2]We adopt the notation $A(\cdot)$ because the most common unbiased estimator of the reward is an unbiased estimator of the advantage function.

sequences), this shift compounds across tokens, making the whole gradient estimation high-variance, which hurts the stability and performance of the training. We provide a rigorous statement of the above intuition and empirical validation in Section D.5.

This motivates alternatives to full importance sampling. A standard way is to clip the importance ratio, yielding a CISPO-style objective (Chen et al., 2025):

$$\mathbb{E}_{y_1 \sim \pi(\cdot|x_1)}\left[\text{clip}\left[\frac{\pi(y_1 \mid x_0)}{\pi_{\text{ref}}(y_1)},\, 1-\varepsilon,\, 1+\varepsilon\right] A(x_0, y_1)\right].$$

Clipping controls variance by truncating rare but high-magnitude ratios, at the cost of a controlled bias that is governed by ε, an additional hyperparameter to tune.

The other extreme is to discard importance weighting entirely, which introduces higher bias but gives the low-variance objective:

$$\nabla \ell^{\text{awr}}_{\text{distill}}(\pi) = \mathbb{E}_{y_1 \sim \pi(\cdot|x_1)}[A(y_1)\, \nabla \log \pi(y_1 \mid x_0)],$$

which resembles advantage-weighted regression (AWR) (Peng et al., 2019; Nair et al., 2020) without the exponential weighting, applied to distillation from feedback-conditioned rollouts.

Empirically, we find that variance dominates bias: setting $\pi_{\text{ref}}(\cdot \mid x_1) = \pi(\cdot \mid x_0)$ (note that this is a special case because x_0 is part of x_1), which removes the importance weighting, consistently improves stability and performance compared to using $\pi_{\text{ref}}(\cdot \mid x_1) = \pi(\cdot \mid x_1)$ with full importance correction or the clipped objective. We provide ablation over all variants in Section 5.2. We therefore view mild bias as benign relative to the variance induced by importance sampling in distillation.

Remark 3.1. For clarity, the analysis uses the sequence-level importance weight $W(y_1) := \frac{\pi(y_1|x_0)}{\pi(y_1|x_1)}$. For an autoregressive policy this factorizes exactly as

$$W(y_1) = \prod_{t=1}^{T} r_t, \qquad r_t := \frac{\pi(y_{1,t} \mid x_0, y_{1,<t})}{\pi(y_{1,t} \mid x_1, y_{1,<t})}.$$

Thus, token-level IS simply computes W via per-token ratios and is not an approximation. In contrast, CISPO-style or PPO-style token-level objectives can be viewed as a first-order approximation in the per-token log-ratios $\Delta_t := \log r_t$ via $W = \exp(\sum_t \Delta_t) \approx 1 + \sum_t \Delta_t$ when Δ_t are small. All our experiments use the token-level surrogate following standard practice (Sheng et al., 2024).

Final algorithm. We adopt **(1)** $\pi_{\text{ref}}(\cdot \mid x_1) = \pi(\cdot \mid x_0)$ for AWR-style RL distillation and **(2)** using first-turn mean reward as baseline (Eq. (5)). We summarize the RLTF-SD algorithm in Algorithm 1.

4 Feedback Modeling

Beyond using feedback-conditioned rollouts to improve the policy, we can also treat the critique itself as a supervision signal and explicitly model the feedback provider. This is appealing because feedback c_h is observed at every turn and is far richer than a scalar reward: it pinpoints the mistakes, providing dense token-level gradients on failure rollouts. To leverage the dense feedback signal, we propose Feedback Modeling: *training the policy to predict the feedback itself.*

Feedback-prediction loss. Recall that at each timestep h the feedback provider samples $c_h \sim \mathcal{M}(\cdot \mid x_h, y_h)$. We define a feedback-prediction distribution $p_\pi(c \mid x, y) := \pi(c \mid f_{\text{FeeMol}}(x, y))$, where f_{FeeMol} is a prompt template that elicits critique-style feedback given (x, y); see examples in Section D.1. Using tuples (x_h, y_h, c_h) collected from interaction trajectories, we optimize the cross-entropy objective

$$\ell_{\text{FeeMol}}(\pi) := \mathbb{E}_\pi\left[\sum_{h=0}^{H-1} -\log p_\pi(c_h \mid x_h, y_h)\right]. \tag{6}$$

Note that we treat y_h as constants (i.e., no gradient) so that ℓ_{FeeMol} is pure supervised learning on the feedback tokens, rather than introducing additional credit assignment through the sampling process.

Joint objective with RL. Similar to the self distillation loss, feedback modeling is used as an auxiliary loss in addition to the regular RL objective:

$$\max_\pi \; J_{\text{MultiTurn}}(\pi) - \lambda_{\text{FeeMol}}\, \ell_{\text{FeeMol}}(\pi), \tag{7}$$

Table 1: Comparison of baselines across **reasoning puzzles**, **competition math**, and **creative writing** tasks. We report single-turn accuracy after 2-turn training (i.e., $J_{\mathsf{SingleTurn}}(\pi)$) of the last checkpoint. For the reasoning tasks and `LitBench`, we report the mean@1 accuracy, judged by either verifiable reward or LLM-as-a-judge. For the math tasks, we report the mean@32 accuracy from the last checkpoint from the training. The parentheses denote the training dataset. For `WritingBench`, we follow the official protocol with GPT-4.1-mini as the judge. The accuracy in reasoning and math is normalized between 0 and 1, and the score in creative writing is normalized between 1 and 10. Note that RLTF-SD and RLTF-FM consistently outperform all baselines across tasks.

	Base Model	GRPO Single turn	GRPO Multi turn	Feedback Descent	RLTF-SD	RLTF-FM
Reasoning						
`Knights and Knaves`	0.058	0.373	0.352	0.055	0.802	0.880
`Binary Matrix`	0.001	0.125	0.950	0.005	0.976	0.978
`Shortest Path`	0.034	0.385	0.384	0.035	0.830	0.905
Math						
`MATH500 (DAPO)`	0.376	0.526	0.523	0.415	0.548	0.567
`AIME24 (DAPO)`	0.025	0.058	0.025	0.045	0.088	0.083
`MATH500 (DeepMath)`	0.376	0.558	0.578	0.424	0.598	0.636
`AIME24 (DeepMath)`	0.025	0.042	0.050	0.054	0.058	0.058
Creative Writing						
`LitBench`	4.20	6.83	6.41	8.25	8.80	8.40
`WritingBench`	5.71	5.92	6.29	5.30	6.71	6.39

where $J_{\mathsf{MultiTurn}}(\pi)$ is the multi-turn RL objective (Eq. (1)) and $\lambda_{\mathsf{FeeMol}} \geq 0$ controls the strength of the auxiliary feedback loss. We provide theoretical analysis for RLTF-FM in Section C.

4.1 Test-time scaling via self-feedback

Because $p_\theta(c \mid x, y)$ is produced by the same policy, the model can be run in a "feedback mode" at inference time to generate critiques and perform iterative refinement: sample $y_0 \sim \pi_\theta(\cdot \mid x_0)$, generate $\tilde{c}_0 \sim p_\theta(\cdot \mid x_0, y_0)$, update $x_1 = f(x_0, y_0, \tilde{c}_0)$, and resample $y_1 \sim \pi_\theta(\cdot \mid x_1)$. This enables test-time scaling without requiring a separate learned judge model; the auxiliary training simply makes the policy's self-critique distribution more faithful to the external feedback channel. We further explore this in Section 5.4. The complete training and inference pseudocode is in Algorithm 2.

5 Experiments

Our experiments evaluate our two proposed methods for RL from text feedback: RLTF-SDand RLTF-FM. The goal of our experiments is twofold: (i) quantify how much these components improve performance over standard RL baselines, and (ii) isolate the design choices that make them effective in practice. Concretely, we seek to answer the following research questions:

RQ1: How well do self distillation and feedback modeling work across a wide range of tasks?
RQ2: Which design choices matter for distillation? In particular, does the proposed design choices (use of baseline, advantage weighted regression) consistently outperform their alternatives?
RQ3: How much of the gain remains if we remove rich critiques and provide only a correctness-style signal?
RQ4: How does feedback modeling enable effective *test-time* scaling by generating multiple rounds of self-feedback at inference time?

Experiment setup and baselines. We use `Qwen3-235B-A22B-Instruct-2507` (Yang et al., 2025) to simulate the feedback provider ($\mathcal{M}$), and we use `Llama-3.1-8B-Instruct` (Grattafiori et al., 2024) as the learner. We use early termination (c.f. Section 2) unless otherwise specified. We compare with a comprehensive set of baselines: for reward-only RL, we use GRPO (Shao et al., 2024), with both single-turn ($J_{\mathsf{SingleTurn}}$) and multi-turn ($J_{\mathsf{MultiTurn}}$) objectives; for text feedback aware baselines, we compare with Feedback Descent (Lee et al., 2025), which performs optimization in text space without modifying model weights. We defer the details to Sections D.1 and D.3.

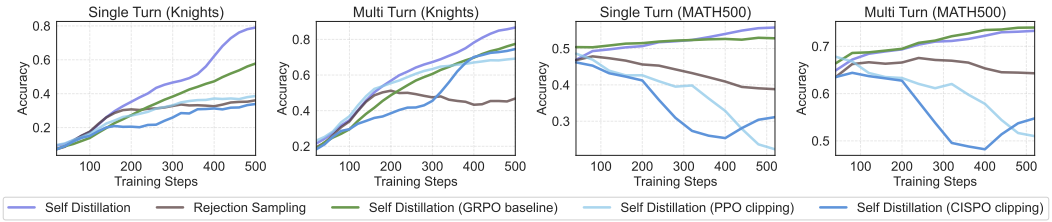

Figure 2: Evaluation curves on `Knights and Knaves` and `MATH500` (trained on `DAPO`) for ablations on self distillation design choices. For each environment: Left: single-turn accuracy; Right: multi-turn accuracy. RLTF-SD (GRPO baseline) denotes using AWR objective with second turn mean baseline. RLTF-SD (PPO clipping) denotes using PPO style clipping on importance weighting with first turn baseline. RLTF-SD (CISPO clipping) denotes using CISPO style clipping on importance weighting with first turn baseline. Note that our proposed design choices consistently outperform the alternatives in both single-turn and multi-turn performance.

5.1 General Results

To investigate **RQ1**, we compare RLTF-SD and RLTF-FM with the baselines across a wide range of tasks, including **reasoning puzzles** (`Knights and Knaves`, `Binary Matrix`, `Shortest Path`) (Stojanovski et al., 2025; Tajwar et al., 2025), **competition math** (training on `DAPO` (Yu et al., 2025) and `Deepmath` (He et al., 2025) and testing on `MATH500` (Hendrycks et al., 2021) and `AIME24`), and **creative writing** (`LitBench` (Fein et al., 2025) and `WritingBench` (Wu et al., 2025)). We defer the details of benchmarks, prompts, and hyperparameters to Sections D.1, D.3 and D.4 respectively.

We focus on the 2-turn setting and compare the final single-turn performance $J_{\mathsf{SingleTurn}}(\pi)$. We summarize the main results in Table 1 and we defer the multi-turn performance and evaluation curves to Section D.6. We observe that both RLTF-SD and RLTF-FM consistently outperform all baselines across tasks, demonstrating the effectiveness of learning from text feedback. Notably, in terms of single-turn performance GRPO with multi-turn training demonstrates similar performance to single-turn training, suggesting that *naively incorporating feedback as additional context is insufficient to internalize its learning signal*. Feedback Descent also underperforms our methods, indicating the importance of parameter space optimization instead of text space optimization. Although both proposed methods outperform the other baselines across the board, the improvement is more significant in the reasoning tasks and `LitBench`, where the train-test distribution mismatch is small, and thus feedback can significantly accelerate learning. Still, incorporating feedback also helps in domains like math and `WritingBench`[3], indicating the generalization of feedback incorporation. To compare RLTF-SD and RLTF-FM, we observe that RLTF-SD outperforms in creative writing tasks where the teacher-student distribution mismatch is small, and RLTF-FM obtains better results in math and reasoning tasks where the feedback is more subjective and the auxiliary prediction loss is easier to optimize.

Case studies. To better understand how text feedback shapes model behavior during training, we qualitatively examine first- and second-turn generations, and show a few examples in Section D.2. These examples demonstrate that feedback can help the model escape local optima that RL can get stuck in (e.g., claiming that all problems are infeasible), correct flawed reasoning chains, and identify arithmetic errors. In this way, text feedback provides targeted, actionable information beyond scalar rewards.

5.2 Ablation on Self Distillation

In this section, we investigate **RQ2** by ablating two major design choices for RLTF-SD: **(i) the use of a baseline for advantage estimation/variance reduction**, where we compare the GRPO style baseline in Eq. (8) and our first-turn baseline in Eq. (5). **(ii) bias-variance tradeoff in importance weighting**, where we compare our AWR-style objective with importance weighting with two different clipping objectives: CISPO style clipping (Chen et al., 2025) and PPO style clipping (Schulman et al., 2017). Finally we also compare with the Rejection Sampling baseline (Scheurer et al., 2023),

[3]For `WritingBench` evaluation, we use the same checkpoint from training on the `LitBench` training set, but `WritingBench` has tasks beyond story writing, which is the only task in `LitBench`.

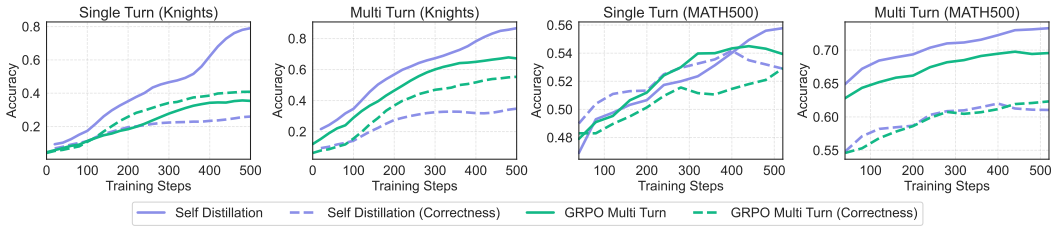

Figure 3: Evaluation curves on `Knights and Knaves` and `MATH500` (trained on `DAPO`) for text feedback vs. correctness-only feedback. We compare single- and multi-turn accuracy on two algorithms: multi-turn GRPO and RLTF-SD. Overall, using text feedback outperforms using correctness-only feedback.

which adds an SFT auxiliary loss to imitate the correct second-turn responses. We perform ablation on the `Knights and Knaves` and math reasoning training with `DAPO` and summarize the results in Figure 2. We observe that introducing importance weighting introduces instability during training, even with different mechanisms of clipping on the importance weighting. Without importance weighting, our first-turn baseline provides significant performance improvement over the regular GRPO-style baseline, indicating the empirical benefit of our design beyond the didactic setting considered in Section 3. Notably, Rejection Sampling also underperforms RLTF-SD and RLTF-SD with GRPO baseline, indicating the benefit of variance reduction via baselines.

5.3 Ablation on Feedback

To answer **RQ3**, we compare to a correctness-only version of RLTF-SD that does not use text feedback and only provides a correctness signal. Specifically, we replace the judge critique after the first turn with the sentence `"Your previous answer was {correct/incorrect}"`. Figure 3 shows the performance with correctness-only feedback on 1) multi-turn GRPO, and 2) RLTF-SD. We find that the correctness-only baseline does not perform well compared to RLTF-SD, indicating that semantically rich text feedback is critical. One notable exception is the single-turn `Knights and Knaves` accuracy using multi-turn GRPO. Without distillation, neither text feedback nor correctness-only feedback can significantly influence the model's first-turn response, so there is little difference in this setting.

5.4 Test-time Scaling of Feedback Modeling

Finally, for **RQ4**, we investigate the test-time scaling ability of RLTF-FM by generating multiple rounds of self-feedback at inference time. Specifically, we evaluate the model trained with RLTF-FM on `Knights and Knaves` and `MATH500` (trained with `DAPO`) by allowing it to generate up to 5 rounds of generation with self-feedback at inference time. We introduce an additional baseline where we use RL to improve model's self-critique using second-turn reward (Algorithm 4), and we disable early termination during the training under this setting. We summarize the results in Figure 4. We make the following observations: first, using RL for learning self-critique is not sufficient: in the math experiment, we observe that GRPO with and without self-critique training achieve similar test-time improvement. Second, adding RLTF-FM loss in addition to self-critique RL training brings significant test-time improvement. Third, the benefit of RLTF-FM is mainly in terms of the magnitude of improvement, not in terms of the number of rounds of improvement. The test-time improvement saturates after a handful of rounds, but this is expected and corroborates with the self-improvement literature (Huang et al., 2023; Song et al., 2024b).

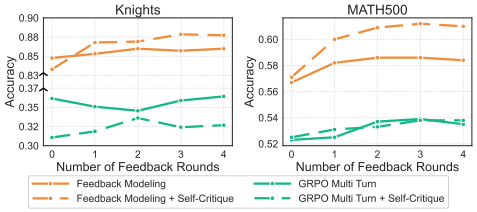

Figure 4: Test-time scaling results on `Knights and Knaves` and `MATH500` (trained on `DAPO`). We allow the model to generate multiple rounds of self-feedback at inference time (x-axis). We compare RLTF-FM with multi-turn scalar-based RL, and the dashed line ("+ Self-Critique") denotes further using RL to improve the self-critique during training (Algorithm 4). We use a skipped y-axis for the left plot for clarity.

6 Related Work

Learning from text feedback. A well-studied area of human-robot interaction explores learning from natural language corrections (Broad et al., 2017; Sharma et al., 2022; Bucker et al., 2022; Liu et al., 2023a; Cui et al., 2023; Lynch et al., 2023; Liang et al., 2024; Shi et al., 2024). In these approaches, humans provide corrections such as "move a bit to the left," grounded in the robot's perception and action space and used to update policies or value functions. Zhao et al. (2026a) incorporates text feedback for image generation by prompting VLMs to provide critiques of generated images. We study learning from text feedback in the context of RL for LLM reasoning (Shao et al., 2024; Guo et al., 2025a; Hu et al., 2025), which typically relies on a single scalar reward. In contrast, learning directly from text feedback preserves semantic structure and compositionality. The theoretical benefit of learning from text feedback has been shown in Pukdee et al. (2023); Xu et al. (2025). Feng et al. (2024); Hong et al. (2025); Zhang et al. (2025c); Yang et al. (2026) study how to incorporate self-critiques into the model (e.g., via policy and value distillation), but the setting differs in that they do not assume access to external text feedback. Another class of methods (Chang et al., 2023; Amani et al., 2025; Li et al., 2025; Zhang et al., 2025b) bridges SFT and RL by revealing partial prefixes of an expert solution to guide RL training. Wang et al. (2025b) converts text feedback to denser span-level rewards, but this ultimately collapses the text into the same order of numerical signals as regular RL. Furthermore, several works (Madaan et al., 2023; Cheng et al., 2024; Yuksekgonul et al., 2024; Lee et al., 2025) have also proposed learning from text feedback by propagating minimal subgraphs, or performing optimization in text space. Finally, in the spirit of goal relabeling (Andrychowicz et al., 2017), feedback-conditioned policies (Liu et al., 2023b; Zhang et al., 2023; Luo et al., 2025) use feedback as a goal in hindsight rather than an intermediate step in a multi-turn interaction.

LLM distillation. In knowledge distillation (Hinton et al., 2015; Kim & Rush, 2016; Sanh et al., 2019), a student model aims to mimic a teacher model's soft probability distribution. On-policy distillation (Agarwal et al., 2024; Xu et al., 2024b; Gu et al., 2023; Lu & Lab, 2025; Xiao et al., 2026; Yang et al., 2025) trains the student on its own generations instead of the teacher's generations. In self-distillation (Askell et al., 2021; Snell et al., 2022; Choi et al., 2022; Kujanpää et al., 2025; Mitra & Ulukus, 2025; Eyuboglu et al., 2025; Caccia et al., 2025), a student model learns from a teacher that has access to privileged information through its prompt. The teacher and student are typically the same base model; the teacher is not inherently more capable, but instead benefits from additional context embedded in the prompt. Prior work has explored self-distillation across a range of applications, including alignment (Askell et al., 2021), instruction following (Snell et al., 2022), and persona-conditioned dialogue (Choi et al., 2022). Kujanpää et al. (2025); Eyuboglu et al. (2025); Caccia et al. (2025); Qu et al. (2025) study how models can learn from unstructured, free-form documents via prompt distillation. Jayalath et al. (2025) synthesizes a target for distillation by combining multiple samples from the model. Mitra & Ulukus (2025) apply self-distillation to reasoning tasks, via a teacher model with access to both correct and incorrect solutions. Zhao et al. (2026b); Shenfeld et al. (2026); Hübotter et al. (2026) are concurrent works in this space; their settings differ as the teacher provides demonstrations of successful attempts instead of feedback on the model's generations. Hübotter et al. (2026) also studies text feedback from interpreters (e.g., runtime errors), which, however, is available at test time, and the proposed approaches do not directly optimize a reward-based objective.

LLM world models. World modeling (Sutton, 1991; Ha & Schmidhuber, 2018; Hafner et al., 2020a;b; 2023) has long been used to improve the sample efficiency of RL. An agent learns to predict future states and rewards given the current state and action, and this internal model enables planning through imagined rollouts rather than direct interaction with the environment. More recently, this idea has been adapted to LLMs (Gu et al., 2024; Guo et al., 2025b; Chae et al., 2024; Hao et al., 2023). In this direction, Zhang et al. (2025a) proposed to have LLMs learn from their own collected interaction data ("early experience"), via an implicit world modeling strategy, which uses next-state prediction to learn the environment dynamics. Copet et al. (2025) released Code World Model (CWM), a 32-billion-parameter LLM trained on large amounts of state-action pairs of Python interpreter traces and interactions with Docker environments. Some works have studied learning to model text feedback ("forward prediction") in the context of dialogue generation (Weston, 2016; Li et al., 2016) and detecting harmful content (Xu et al., 2024a); in our work, we study how to combine feedback modeling and reinforcement learning.

Multi-turn RL. In the context of LLMs, generating a complete response and receiving a reward signal without intermediate intervention is often sufficient as there is no need for interaction. However, as LLMs are increasingly deployed as autonomous agents, the need for multi-turn RL has grown significantly. Recently, multi-turn RL (Zhou et al., 2024; Kumar et al., 2024; Abdulhai et al., 2023) has been studied more extensively for agentic settings where interacting with an external environment is beneficial, such as interacting with the terminal (Liu et al., 2023c) or the Internet (Zhou et al., 2023). Several methods (Wang et al., 2025a; Ji et al., 2024; Zhou et al., 2025) have been developed to improve sample complexity and long-horizon performance for multi-turn RL. In our work, this "environment" is the feedback provider, which impacts the model's second-turn generation by critiquing its first-turn response.

7 Conclusion and Discussion

We study RL from text feedback, addressing the sparsity of scalar rewards while providing a scalable alternative to expert demonstration. Our two methods, Self Distillation (RLTF-SD) and Feedback Modeling (RLTF-FM), enjoy favorable theoretical properties and demonstrate strong empirical performance across reasoning, math, and creative writing tasks. As text feedback becomes increasingly abundant through human-AI interaction, we see RL from text feedback as a natural next step beyond reward optimization. Several limitations suggest directions for future work. First, real-world feedback may be noisy or subjective, likely requiring data curation and filtering. Second, while our methods generalize to arbitrary horizons, truly long-horizon feedback interaction may require techniques such as summarization to address distribution shift and context limits. Third, our theory focuses on representation learning near the base policy's distribution; a fully end-to-end analysis would strengthen understanding of feedback modeling. Finally, exploring interplay with other fine-grained supervision methods, such as process reward models (Lightman et al., 2023) is a promising direction.

References

Abdulhai, M., White, I., Snell, C., Sun, C., Hong, J., Zhai, Y., Xu, K., and Levine, S. Lmrl gym: Benchmarks for multi-turn reinforcement learning with language models. *arXiv preprint arXiv:2311.18232*, 2023.

Agarwal, R., Vieillard, N., Zhou, Y., Stanczyk, P., Garea, S. R., Geist, M., and Bachem, O. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

Agrawal, L. A., Tan, S., Soylu, D., Ziems, N., Khare, R., Opsahl-Ong, K., Singhvi, A., Shandilya, H., Ryan, M. J., Jiang, M., et al. Gepa: Reflective prompt evolution can outperform reinforcement learning. *arXiv preprint arXiv:2507.19457*, 2025.

Amani, M. H., Lotfi, A., Baldwin, N. M., Bengio, S., Farajtabar, M., Abbe, E., and West, R. Rl for reasoning by adaptively revealing rationales. *arXiv preprint arXiv:2506.18110*, 2025.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., and Zaremba, W. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., DasSarma, N., et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Ba, J. and Caruana, R. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27, 2014.

Broad, A., Arkin, J., Ratliff, N., Howard, T., and Argall, B. Real-time natural language corrections for assistive robotic manipulators. *The International Journal of Robotics Research*, 36(5-7):684–698, 2017.

Buciluă, C., Caruana, R., and Niculescu-Mizil, A. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.

Bucker, A., Figueredo, L., Haddadin, S., Kapoor, A., Ma, S., Vemprala, S., and Bonatti, R. Latte: Language trajectory transformer. *arXiv preprint arXiv:2208.02918*, 2022.

Caccia, L., Ansell, A., Ponti, E., Vulić, I., and Sordoni, A. Training plug-n-play knowledge modules with deep context distillation. *arXiv preprint arXiv:2503.08727*, 2025.

Chae, H., Kim, N., Ong, K. T.-i., Gwak, M., Song, G., Kim, J., Kim, S., Lee, D., and Yeo, J. Web agents with world models: Learning and leveraging environment dynamics in web navigation. *arXiv preprint arXiv:2410.13232*, 2024.

Chang, J. D., Brantley, K., Ramamurthy, R., Misra, D., and Sun, W. Learning to generate better than your llm. *arXiv preprint arXiv:2306.11816*, 2023.

Chang, J. D., Zhan, W., Oertell, O., Brantley, K., Misra, D., Lee, J. D., and Sun, W. Dataset reset policy optimization for rlhf. *arXiv preprint arXiv:2404.08495*, 2024.

Chen, A., Li, A., Gong, B., Jiang, B., Fei, B., Yang, B., Shan, B., Yu, C., Wang, C., Zhu, C., et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *arXiv preprint arXiv:2506.13585*, 2025.

Cheng, C.-A., Nie, A., and Swaminathan, A. Trace is the next autodiff: Generative optimization with rich feedback, execution traces, and llms. *Advances in Neural Information Processing Systems*, 37: 71596–71642, 2024.

Choi, E., Jo, Y., Jang, J., and Seo, M. Prompt injection: Parameterization of fixed inputs. *arXiv preprint arXiv:2206.11349*, 2022.

Copet, J., Carbonneaux, Q., Cohen, G., Gehring, J., Kahn, J., Kossen, J., Kreuk, F., McMilin, E., Meyer, M., Wei, Y., et al. Cwm: An open-weights llm for research on code generation with world models. *arXiv preprint arXiv:2510.02387*, 2025.

Cui, Y., Karamcheti, S., Palleti, R., Shivakumar, N., Liang, P., and Sadigh, D. No, to the right: Online language corrections for robotic manipulation via shared autonomy. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 93–101, 2023.

Eyuboglu, S., Ehrlich, R., Arora, S., Guha, N., Zinsley, D., Liu, E., Tennien, W., Rudra, A., Zou, J., Mirhoseini, A., et al. Cartridges: Lightweight and general-purpose long context representations via self-study. *arXiv preprint arXiv:2506.06266*, 2025.

Fein, D., Russo, S., Xiang, V., Jolly, K., Rafailov, R., and Haber, N. Litbench: A benchmark and dataset for reliable evaluation of creative writing. *arXiv preprint arXiv:2507.00769*, 2025.

Feng, X., Liu, B., Song, Y., Fu, H., Wan, Z., Koushik, G. A., Hu, Z., Yang, M., Wen, Y., and Wang, J. Natural language reinforcement learning. *arXiv preprint arXiv:2411.14251*, 2024.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Gu, Y., Dong, L., Wei, F., and Huang, M. Minillm: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*, 2023.

Gu, Y., Zhang, K., Ning, Y., Zheng, B., Gou, B., Xue, T., Chang, C., Srivastava, S., Xie, Y., Qi, P., et al. Is your llm secretly a world model of the internet? model-based planning for web agents. *arXiv preprint arXiv:2411.06559*, 2024.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Guo, S., Domingues, O. D., Avalos, R., Courville, A., and Strub, F. Sample, predict, then proceed: Self-verification sampling for tool use of llms. *arXiv preprint arXiv:2506.02918*, 2025b.

Ha, D. and Schmidhuber, J. Recurrent world models facilitate policy evolution. *Advances in Neural Information Processing Systems*, 2018.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020a.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020b.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models. *arXiv:2301.04104*, 2023.

Hao, S., Gu, Y., Ma, H., Hong, J., Wang, Z., Wang, D., and Hu, Z. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8154–8173, 2023.

He, Z., Liang, T., Xu, J., Liu, Q., Chen, X., Wang, Y., Song, L., Yu, D., Liang, Z., Wang, W., Zhang, Z., Wang, R., Tu, Z., Mi, H., and Yu, D. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. 2025. URL https://arxiv.org/abs/2504.11456.

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.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hong, J., Liu, K., Ling, Z., Chen, J., and Levine, S. Natural language actor-critic: Scalable off-policy learning in language space. *arXiv preprint arXiv:2512.04601*, 2025.

Hu, J., Zhang, Y., Han, Q., Jiang, D., Zhang, X., and Shum, H.-Y. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

Huang, J., Gu, S., Hou, L., Wu, Y., Wang, X., Yu, H., and Han, J. Large language models can self-improve. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 1051–1068, 2023.

Hübotter, J., Lübeck, F., Behric, L., Baumann, A., Bagatella, M., Marta, D., Hakimi, I., Shenfeld, I., Buening, T. K., Guestrin, C., et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Jayalath, D., Goel, S., Foster, T., Jain, P., Gururangan, S., Zhang, C., Goyal, A., and Schelten, A. Compute as teacher: Turning inference compute into reference-free supervision. *arXiv preprint arXiv:2509.14234*, 2025.

Ji, K., Liu, G., Dai, N., Yang, Q., Zheng, R., Wu, Z., Dun, C., Gu, Q., and Yan, L. Enhancing multi-step reasoning abilities of language models through direct q-function optimization. *arXiv preprint arXiv:2410.09302*, 2024.

Jin, Y., Yang, Z., and Wang, Z. Is pessimism provably efficient for offline rl? In *International conference on machine learning*, pp. 5084–5096. PMLR, 2021.

Kim, Y. and Rush, A. M. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 1317–1327, 2016.

Kujanpää, K., Marttinen, P., Valpola, H., and Ilin, A. Efficient knowledge injection in llms via self-distillation. *Transactions on Machine Learning Research*, 2025.

Kumar, A., Zhuang, V., Agarwal, R., Su, Y., Co-Reyes, J. D., Singh, A., Baumli, K., Iqbal, S., Bishop, C., Roelofs, R., et al. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*, 2024.

Lab, T. M. Tinker, 2025. URL https://thinkingmachines.ai/tinker/.

Lee, Y., Boen, J., and Finn, C. Feedback descent: Open-ended text optimization via pairwise comparison. *arXiv preprint arXiv:2511.07919*, 2025.

Li, J., Miller, A. H., Chopra, S., Ranzato, M., and Weston, J. Dialogue learning with human-in-the-loop. *arXiv preprint arXiv:1611.09823*, 2016.

Li, J., Lin, H., Lu, H., Wen, K., Yang, Z., Gao, J., Wu, Y., and Zhang, J. Questa: Expanding reasoning capacity in llms via question augmentation. *arXiv preprint arXiv:2507.13266*, 2025.

Liang, J., Xia, F., Yu, W., Zeng, A., Arenas, M. G., Attarian, M., Bauza, M., Bennice, M., Bewley, A., Dostmohamed, A., et al. Learning to learn faster from human feedback with language model predictive control. *arXiv preprint arXiv:2402.11450*, 2024.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Liu, H., Chen, A., Zhu, Y., Swaminathan, A., Kolobov, A., and Cheng, C.-A. Interactive robot learning from verbal correction. *arXiv preprint arXiv:2310.17555*, 2023a.

Liu, H., Sferrazza, C., and Abbeel, P. Chain of hindsight aligns language models with feedback. *arXiv preprint arXiv:2302.02676*, 2023b.

Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023c.

Lu, K. and Lab, T. M. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Luo, R., Liu, Z., Liu, X., Du, C., Lin, M., Chen, W., Lu, W., and Pang, T. Language models can learn from verbal feedback without scalar rewards. *arXiv preprint arXiv:2509.22638*, 2025.

Lynch, C., Wahid, A., Tompson, J., Ding, T., Betker, J., Baruch, R., Armstrong, T., and Florence, P. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594, 2023.

Mitra, P. and Ulukus, S. Semantic soft bootstrapping: Long context reasoning in llms without reinforcement learning. *arXiv preprint arXiv:2512.05105*, 2025.

Nair, A., Gupta, A., Dalal, M., and Levine, S. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

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.

Peng, X. B., Kumar, A., Zhang, G., and Levine, S. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Pukdee, R., Sam, D., Kolter, J. Z., Balcan, M.-F. F., and Ravikumar, P. Learning with explanation constraints. *Advances in neural information processing systems*, 36:49883–49926, 2023.

Qu, Y., Setlur, A., Smith, V., Salakhutdinov, R., and Kumar, A. How to explore to scale rl training of llms on hard problems? urlhttps://blog.ml.cmu.edu/2025/11/26/how-to-explore-to-scale-rl-training-of-llms-on-hard-problems, 2025. CMU MLD Blog.

Ross, S., Gordon, G., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Scheurer, J., Campos, J. A., Korbak, T., Chan, J. S., Chen, A., Cho, K., and Perez, E. Training language models with language feedback at scale. *arXiv preprint arXiv:2303.16755*, 2023.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

Shani, L., Rosenberg, A., Cassel, A., Lang, O., Calandriello, D., Zipori, A., Noga, H., Keller, O., Piot, B., Szpektor, I., et al. Multi-turn reinforcement learning with preference human feedback. *Advances in Neural Information Processing Systems*, 37:118953–118993, 2024.

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

Sharma, P., Sundaralingam, B., Blukis, V., Paxton, C., Hermans, T., Torralba, A., Andreas, J., and Fox, D. Correcting robot plans with natural language feedback. *arXiv preprint arXiv:2204.05186*, 2022.

Shenfeld, I., Damani, M., Hübotter, J., and Agrawal, P. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

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. *arXiv preprint arXiv: 2409.19256*, 2024.

Shi, L. X., Hu, Z., Zhao, T. Z., Sharma, A., Pertsch, K., Luo, J., Levine, S., and Finn, C. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024.

Snell, C., Klein, D., and Zhong, R. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Song, Y., Swamy, G., Singh, A., Bagnell, J., and Sun, W. The importance of online data: Understanding preference fine-tuning via coverage. *Advances in Neural Information Processing Systems*, 37: 12243–12270, 2024a.

Song, Y., Zhang, H., Eisenach, C., Kakade, S., Foster, D., and Ghai, U. Mind the gap: Examining the self-improvement capabilities of large language models. *arXiv preprint arXiv:2412.02674*, 2024b.

Stojanovski, Z., Stanley, O., Sharratt, J., Jones, R., Adefioye, A., Kaddour, J., and Köpf, A. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2505.24760*, 2025.

Sutton, R. S. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

Tajwar, F., Jiang, Y., Thankaraj, A., Rahman, S. S., Kolter, J. Z., Schneider, J., and Salakhutdinov, R. Training a generally curious agent. *arXiv preprint arXiv:2502.17543*, 2025.

Tan, S., Luo, M., Cai, C., Venkat, T., Montgomery, K., Hao, A., Wu, T., Balyan, A., Roongta, M., Wang, C., Li, L. E., Popa, R. A., and Stoica, I. rllm: A framework for post-training language agents, 2025. Notion Blog.

Team, T. D., Li, B., Zhang, B., Zhang, D., Huang, F., Li, G., Chen, G., Yin, H., Wu, J., Zhou, J., et al. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025.

Uehara, M. and Sun, W. Pessimistic model-based offline reinforcement learning under partial coverage. *arXiv preprint arXiv:2107.06226*, 2021.

Wang, H., Hao, S., Dong, H., Zhang, S., Bao, Y., Yang, Z., and Wu, Y. Offline reinforcement learning for llm multi-step reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8881–8893, 2025a.

Wang, H., Wang, L., Zhang, C., Mao, T., Qin, S., Lin, Q., Rajmohan, S., and Zhang, D. Text2grad: Reinforcement learning from natural language feedback. *arXiv preprint arXiv:2505.22338*, 2025b.

Weston, J. E. Dialog-based language learning. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf.

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

Wu, Y., Mei, J., Yan, M., Li, C., Lai, S., Ren, Y., Wang, Z., Zhang, J., Wu, M., Jin, Q., et al. Writingbench: A comprehensive benchmark for generative writing. *arXiv preprint arXiv:2503.05244*, 2025.

Xiao, B., Xia, B., Yang, B., Gao, B., Shen, B., Zhang, C., He, C., Lou, C., Luo, F., Wang, G., et al. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*, 2026.

Xu, W., Cai, D., Zhang, Z., Lam, W., and Shi, S. Reasons to reject? aligning language models with judgments. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12288–12304, 2024a.

Xu, W., Han, R., Wang, Z., Le, L. T., Madeka, D., Li, L., Wang, W. Y., Agarwal, R., Lee, C.-Y., and Pfister, T. Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. *arXiv preprint arXiv:2410.11325*, 2024b.

Xu, W., Nie, A., Zheng, R., Modi, A., Swaminathan, A., and Cheng, C.-A. Provably learning from language feedback. *arXiv preprint arXiv:2506.10341*, 2025.

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.

Yang, M. Y. R., Bai, H., Wu, I., Yang, G., Setlur, A., and Kumar, A. Int: Self-proposed interventions enable credit assignment in llm reasoning, 2026. URL https://arxiv.org/abs/2601.14209.

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

Yuksekgonul, M., Bianchi, F., Boen, J., Liu, S., Huang, Z., Guestrin, C., and Zou, J. Textgrad: Automatic" differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024.

Zeng, G., Zhou, Z., Arora, D., and Zanette, A. Shrinking the variance: Shrinkage baselines for reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2511.03710*, 2025.

Zhang, K., Chen, X., Liu, B., Xue, T., Liao, Z., Liu, Z., Wang, X., Ning, Y., Chen, Z., Fu, X., et al. Agent learning via early experience. *arXiv preprint arXiv:2510.08558*, 2025a.

Zhang, T., Liu, F., Wong, J., Abbeel, P., and Gonzalez, J. E. The wisdom of hindsight makes language models better instruction followers. In *International Conference on Machine Learning*, pp. 41414–41428. PMLR, 2023.

Zhang, X., Huang, Z., Li, Y., Ni, C., Chen, J., and Oymak, S. Bread: Branched rollouts from expert anchors bridge sft & rl for reasoning. *arXiv preprint arXiv:2506.17211*, 2025b.

Zhang, X., Sun, H., Zhang, Y., Feng, K., Lu, C., Yang, C., and Meng, H. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback. *arXiv preprint arXiv:2506.03106*, 2025c.

Zhao, H., Chen, H., Guo, Y., Winata, G. I., Ou, T., Huang, Z., Yao, D. D., and Tang, W. Rpo: Fine-tuning visual generative models via rich vision-language preferences, 2026a. URL https://arxiv.org/abs/2503.11720.

Zhao, S., Xie, Z., Liu, M., Huang, J., Pang, G., Chen, F., and Grover, A. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026b.

Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

Zhou, Y., Zanette, A., Pan, J., Levine, S., and Kumar, A. Archer: Training language model agents via hierarchical multi-turn rl. *arXiv preprint arXiv:2402.19446*, 2024.

Zhou, Y., Jiang, S., Tian, Y., Weston, J., Levine, S., Sukhbaatar, S., and Li, X. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*, 2025.

A OMITTED ALGORITHMS

Algorithm 1 Self Distillation

require Initial policy π_θ; group size N; learning rate η; steps T, optimizer OPT.

for $t = 1, 2, \ldots, T$ **do**
  Sample a minibatch of prompts $\{x_0^b\}_{b=1}^B \sim \rho$
  for $b = 1, 2, \ldots, B$ **do**
    for $i = 1, 2, \ldots, N$ **do**
      Sample first-turn output $y_0^{i,b} \sim \pi_\theta(\cdot \mid x_0^b)$
      Obtain feedback $c_0^{i,b} \sim \mathcal{M}(x_0^b, y_0^{i,b})$
      Form second-turn state $x_1^{i,b} \leftarrow f(x_0^b, y_0^{i,b}, c_0^{i,b})$
      Sample second-turn output $y_1^{i,b} \sim \pi_\theta(\cdot \mid x_1^{i,b})$
      Get rewards $r_0^{i,b} \leftarrow R(x_0^b, y_0^{i,b})$ and $r_1^{i,b} \leftarrow R(x_0^b, y_1^{i,b})$
      Compute return $R^{i,b} \leftarrow r_0^{i,b} + \gamma r_1^{i,b}$
    Compute baselines $b^{(0)} \leftarrow \frac{1}{N}\sum_{i=1}^N r_0^{i,b}$, $b^{(R)} \leftarrow \frac{1}{N}\sum_{i=1}^N R^{i,b}$, and $b^{(1)} \leftarrow \frac{1}{N}\sum_{i=1}^N r_1^{i,b}$
    Compute self distillation advantages $A^{i,b} \leftarrow r_1^{i,b} - b^{(0)}$ for all $i \in [N]$
    Compute first turn RL advantages $A_{\mathsf{RL},0}^{i,b} \leftarrow R^{i,b} - b^{(R)}$ for all $i \in [N]$
    Compute second turn RL advantages $A_{\mathsf{RL},1}^{i,b} \leftarrow r_1^{i,b} - b^{(1)}$ for all $i \in [N]$
    Form self distillation gradient estimate

$$\widehat{g}^b \leftarrow \frac{1}{N}\sum_{i=1}^N A^{i,b}\, \nabla_\theta \log \pi_\theta(y_1^{i,b} \mid x_0^b)$$

    Form RL gradient estimate

$$\widehat{g}_{\mathsf{RL}}^b \leftarrow \frac{1}{N}\sum_{i=1}^N \Big[A_{\mathsf{RL},0}^{i,b}\, \nabla_\theta \log \pi_\theta(y_0^{i,b} \mid x_0^b) + A_{\mathsf{RL},1}^{i,b}\, \nabla_\theta \log \pi_\theta(y_1^{i,b} \mid x_1^{i,b})\Big]$$

    Update policy: $\theta \leftarrow \mathrm{OPT}(\theta, \eta, \widehat{g}^b + \widehat{g}_{\mathsf{RL}}^b)$

return π_θ

Algorithm 2 Feedback Modeling with Test-time Self-Feedback

require Initial policy π_θ; number of self-critique steps H.

Sample prompt $x_0 \sim \rho$
for $h = 1, 2, \ldots, H$ **do**
  Sample output $y_h \sim \pi_\theta(\cdot \mid x_{h-1})$
  Generate self-critique $\tilde{c}_h \sim p_\theta(\cdot \mid x_{h-1}, y_h)$
  Form next state $x_h \leftarrow f(x_{h-1}, y_h, \tilde{c}_h)$
return final output y_H

Algorithm 3 Feedback Modeling

require Initial policy π_θ; group size N; learning rate η; steps T, optimizer OPT.

for $t = 1, 2, \ldots, T$ **do**
  Sample a minibatch of prompts $\{x_0^b\}_{b=1}^B \sim \rho$
  for $b = 1, 2, \ldots, B$ **do**
    for $i = 1, 2, \ldots, N$ **do**
      Sample first-turn output $y_0^{i,b} \sim \pi_\theta(\cdot \mid x_0^b)$
      Obtain feedback $c_0^{i,b} \sim \mathcal{M}(x_0^b, y_0^{i,b})$
      Form second-turn state $x_1^{i,b} \leftarrow f(x_0^b, y_0^{i,b}, c_0^{i,b})$
      Sample second-turn output $y_1^{i,b} \sim \pi_\theta(\cdot \mid x_1^{i,b})$
      Get rewards $r_0^{i,b} \leftarrow R(x_0^b, y_0^{i,b})$ and $r_1^{i,b} \leftarrow R(x_0^b, y_1^{i,b})$
      Compute return $R^{i,b} \leftarrow r_0^{i,b} + \gamma r_1^{i,b}$
    Compute baselines $b^{(R)} \leftarrow \frac{1}{N}\sum_{i=1}^N R^{i,b}$, and $b^{(1)} \leftarrow \frac{1}{N}\sum_{i=1}^N r_1^{i,b}$
    Compute first turn RL advantages $A_{\mathsf{RL},0}^{i,b} \leftarrow R^{i,b} - b^{(R)}$ for all $i \in [N]$
    Compute second turn RL advantages $A_{\mathsf{RL},1}^{i,b} \leftarrow r_1^{i,b} - b^{(1)}$ for all $i \in [N]$
    Form feedback modeling gradient estimate

$$\widehat{g}^b \leftarrow \frac{1}{N}\sum_{i=1}^N \nabla_\theta \log \pi_\theta\Big(c_0^{i,b} \mid f_{\mathsf{FeeMol}}(x_0^b, y_0^{i,b})\Big)$$

    Form RL gradient estimate

$$\widehat{g}_{\mathsf{RL}}^b \leftarrow \frac{1}{N}\sum_{i=1}^N \Big[A_{\mathsf{RL},0}^{i,b}\, \nabla_\theta \log \pi_\theta(y_0^{i,b} \mid x_0^b) + A_{\mathsf{RL},1}^{i,b}\, \nabla_\theta \log \pi_\theta(y_1^{i,b} \mid x_1^{i,b})\Big]$$

    Update policy: $\theta \leftarrow \mathrm{OPT}(\theta, \eta, \widehat{g}^b + \widehat{g}_{\mathsf{RL}}^b)$
return π_θ

Algorithm 4 Feedback Modeling with Self-Critique

require Initial policy π_θ; group size N; learning rate η; steps T, optimizer OPT.

for $t = 1, 2, \ldots, T$ **do**
  Sample a minibatch of prompts $\{x_0^b\}_{b=1}^B \sim \rho$
  for $b = 1, 2, \ldots, B$ **do**
    for $i = 1, 2, \ldots, N$ **do**
      Sample first-turn output $y_0^{i,b} \sim \pi_\theta(\cdot \mid x_0^b)$
      Obtain feedback $c_0^{i,b} \sim \mathcal{M}(x_0^b, y_0^{i,b})$
      Sample self-critique $\tilde{c}_0^{i,b} \sim p_\theta(\cdot \mid f_{\mathsf{FeeMol}}(x_0^b, y_0^{i,b}))$
      Form second-turn state $x_1^{i,b} \leftarrow f(x_0^b, y_0^{i,b}, c_0^{i,b}), \tilde{x}_1^{i,b} \leftarrow f(x_0^b, y_0^{i,b}, \tilde{c}_0^{i,b})$
      Sample second-turn output $y_1^{i,b} \sim \pi_\theta(\cdot \mid x_1^{i,b}), \tilde{y}_1^{i,b} \sim \pi_\theta(\cdot \mid \tilde{x}_1^{i,b})$
      Get rewards $r_0^{i,b} \leftarrow R(x_0^b, y_0^{i,b})$, $r_1^{i,b} \leftarrow R(x_0^b, y_1^{i,b})$ and $\tilde{r}_1^{i,b} \leftarrow R(x_0^b, \tilde{y}_1^{i,b})$
      Compute return $R^{i,b} \leftarrow r_0^{i,b} + \frac{\gamma}{2}\left(r_1^{i,b} + \tilde{r}_1^{i,b}\right)$
    Compute baselines $b^{(R)} \leftarrow \frac{1}{N}\sum_{i=1}^N R^{i,b}$, and $b^{(1)} \leftarrow \frac{1}{N}\sum_{i=1}^N r_1^{i,b}, \tilde{b}^{(1)} \leftarrow \frac{1}{N}\sum_{i=1}^N \tilde{r}_1^{i,b}$
    Compute first turn RL advantages $A_{\mathsf{RL},0}^{i,b} \leftarrow R^{i,b} - b^{(R)}$ for all $i \in [N]$
    Compute second turn RL advantages $A_{\mathsf{RL},1}^{i,b} \leftarrow r_1^{i,b} - b^{(1)}, \tilde{A}_{\mathsf{RL},1}^{i,b} \leftarrow \tilde{r}_1^{i,b} - \tilde{b}^{(1)}$ for all $i \in [N]$
    Form feedback modeling gradient estimate

$$\widehat{g}^b \leftarrow \frac{1}{N}\sum_{i=1}^N \nabla_\theta \log \pi_\theta\Big(c_0^{i,b} \mid f_{\mathsf{FeeMol}}(x_0^b, y_0^{i,b})\Big)$$

    Form RL gradient estimate

$$\widehat{g}_{\mathsf{RL}}^b \leftarrow \frac{1}{N}\sum_{i=1}^N \Big[A_{\mathsf{RL},0}^{i,b}\, \nabla_\theta \log \pi_\theta(y_0^{i,b} \mid x_0^b) + A_{\mathsf{RL},1}^{i,b}\, \nabla_\theta \log \pi_\theta(y_1^{i,b} \mid x_1^{i,b}) + \tilde{A}_{\mathsf{RL},1}^{i,b}\, \nabla_\theta \log \pi_\theta(\tilde{c}^{i,b} \mid f_{\mathsf{FeeMol}}(x_0^b, y_0^{i,b}))\Big]$$

    Update policy: $\theta \leftarrow \mathrm{OPT}(\theta, \eta, \widehat{g}^b + \widehat{g}_{\mathsf{RL}}^b)$

return π_θ

B THEORY RESULTS FROM SECTION 3

A natural choice is to use the GRPO-style (Guo et al., 2025a) group-mean baseline computed from second-turn rewards, as this is the standard baseline in multi-turn LLM RL (Team et al., 2025; Tan et al., 2025). Concretely, for each prompt x_0, given $\{(y_0^i, y_1^i)\}_{i=1}^N$ where $y_0^i \sim \pi(\cdot \mid x_0)$ and $y_1^i \sim \pi(\cdot \mid x_1^i)$ with $x_1^i = f(x_0, y_0^i, c_0^i)$, the advantage estimator is defined as

$$A_i^{(1)} \;:=\; R(x_0, y_1^i) \;-\; \frac{1}{N}\sum_{j=1}^{N} R(x_0, y_1^j). \tag{8}$$

In the setting of self-distillation (Eq. (3)), with importance-sampling correction $\pi_{\text{ref}}(\cdot) = \pi(\cdot \mid x_1)$, this yields an unbiased gradient up to a constant multiplicative factor:

$$\mathbb{E}\left[\frac{\text{sg}[\pi(y_1^i \mid x_0)]}{\text{sg}[\pi(y_1^i \mid x_1^i)]} A_i^{(1)} \nabla \log \pi(y_1^i \mid x_0)\right] = \frac{N-1}{N} \nabla J(\pi).$$

Note that this bias can be removed by using a leave-one-out baseline $\frac{1}{N-1}\sum_{j\neq i} R(x_0, y_1^j)$ instead of the in-sample mean, but generally this does not matter in practice as the optimizer is agnostic to constant scaling of the gradient.

However, this baseline has a more serious issue: **gradient-signal collapse under second-turn mean baselines**. A second-turn group-mean baseline centers rewards using the same second-turn samples: this can be unbiased in expectation, but it exhibits a *point-wise* degeneracy: whenever the group rewards are (nearly) constant, the centered reward-estimations vanish and the update is exactly (or approximately) zero. This failure mode is not rare in the multi-turn setting. Let $R(x_0, y_1) \in \{0, 1\}$ with second-turn success probability p_1 for a fixed prompt x_0. Then a second-turn mean baseline yields an exactly zero update whenever the group is constant, which occurs with probability $p_1^N + (1-p_1)^N$. In particular, when feedback makes the second-turn policy highly reliable ($p_1 \to 1$), the probability of a non-zero update scales as $1 - p_1^N \approx N(1-p_1)$, so there is no learning signal for the first turn even though the teacher is consistently correct (at the second turn).

B.1 PROPERTIES OF BASELINES

Setup and notation. Fix a prompt x_0. For $i = 1, \ldots, N$, we sample

$$y_0^i \sim \pi(\cdot \mid x_0), \qquad x_1^i = f(x_0, y_0^i, c_0^i), \qquad y_1^i \sim \pi(\cdot \mid x_1^i),$$

and define rewards $r_0^i := r(x_0, y_0^i)$ and $r_1^i := r(x_0, y_1^i)$. We consider the importance-corrected score-function estimator for the single-turn objective

$$J(\pi) = \mathbb{E}_{x_0\sim\rho}\big[\mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0, y)]\big].$$

For a fixed x_0, define

$$g_i := \frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\, r_1^i\, \nabla \log \pi(y_1^i \mid x_0).$$

Under the standard support condition, the expectation of g_i equals the true single-turn policy gradient at x_0,

$$\mathbb{E}[g_i \mid x_0] \;=\; \nabla\, \mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0, y)].$$

Proposition B.1 (In-sample second-turn group-mean baseline yields $(1 - \frac{1}{N})$ shrinkage)**.** *For a fixed x_0, define the in-sample second-turn mean baseline*

$$\bar{r}_1 \;:=\; \frac{1}{N}\sum_{j=1}^{N} r_1^j, \qquad A_i^{(1)} \;:=\; r_1^i - \bar{r}_1.$$

Consider the importance-corrected gradient estimator

$$\hat{G}^{(2)} \;:=\; \frac{1}{N}\sum_{i=1}^{N} \frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\, A_i^{(1)}\, \nabla \log \pi(y_1^i \mid x_0).$$

Then, conditioning on x_0,

$$\mathbb{E}\Big[\hat{G}^{(2)} \mid x_0\Big] \;=\; \left[1-\frac{1}{N}\right]\nabla\,\mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0,y)].$$

Equivalently, the in-sample second-turn mean baseline introduces a multiplicative shrinkage factor $(1-\frac{1}{N})$ in expectation.

Proof. Fix x_0. By exchangeability it suffices to analyze a single index i and then take the average. Write

$$A_i^{(1)} \;=\; r_1^i - \frac{1}{N}\sum_{j=1}^{N} r_1^j \;=\; \left[1-\frac{1}{N}\right]r_1^i - \frac{1}{N}\sum_{j\neq i} r_1^j.$$

Hence

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,A_i^{(1)}\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right] = \left[1-\frac{1}{N}\right]\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,r_1^i\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right] - \frac{1}{N}\sum_{j\neq i}\mathbb{E}\left[r_1^j\cdot\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right].$$

For $j \neq i$, the random variable r_1^j depends only on (y_0^j, x_1^j, y_1^j) and is independent of (y_0^i, x_1^i, y_1^i) conditioned on x_0 (since the N rollouts are i.i.d. given x_0). Therefore the cross term factors:

$$\mathbb{E}\left[r_1^j\cdot\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right] = \mathbb{E}[r_1^j \mid x_0]\,\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right].$$

Next, for the second factor, note that

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right] = \mathbb{E}_{y\sim\pi(\cdot|x_0)}[\nabla\log\pi(y \mid x_0)] = \nabla\int\pi(y \mid x_0)\,dy = 0,$$

where the first equality is the standard importance-sampling identity under $\pi_{\mathrm{ref}}(\cdot) = \pi(\cdot \mid x_1^i)$. Hence every cross term vanishes. We conclude

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,A_i^{(1)}\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right] = \left[1-\frac{1}{N}\right]\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,r_1^i\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right].$$

Finally, by the unbiasedness of the importance-corrected estimator with $A = r_1$,

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,r_1^i\,\nabla\log\pi(y_1^i \mid x_0) \mid x_0\right] = \nabla\,\mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0,y)].$$

Averaging over i gives the claimed result. □

Remark. Replacing $\bar{r}_1$ by the leave-one-out baseline $\bar{r}_{1,-i} = \frac{1}{N-1}\sum_{j\neq i} r_1^j$ eliminates the self-coupling term and yields $\mathbb{E}[\hat{G}^{(2)} \mid x_0] = \nabla\,\mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0,y)]$.

Proposition B.2 (First-turn group-mean baseline is unbiased (with IS correction))**.** *For a fixed x_0, define the first-turn mean baseline*

$$\bar{r}_0 \;:=\; \frac{1}{N}\sum_{j=1}^{N} r_0^j, \qquad A_i^{(0)} \;:=\; r_1^i - \bar{r}_0.$$

Consider the importance-corrected gradient estimator

$$\hat{G}^{(1)} \;:=\; \frac{1}{N}\sum_{i=1}^{N}\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,A_i^{(0)}\,\nabla\log\pi(y_1^i \mid x_0).$$

Then, conditioning on x_0,

$$\mathbb{E}\Big[\hat{G}^{(1)} \mid x_0\Big] \;=\; \nabla\,\mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0,y)].$$

In other words, the first-turn mean baseline does not introduce bias in expectation.

Proof. Fix x_0 and an index i. Expanding the expectation,

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,(r_1^i - \bar{r}_0)\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0\right]$$
$$= \mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,r_1^i\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0\right] - \mathbb{E}\left[\bar{r}_0 \cdot \frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0\right].$$

The first term equals the desired gradient by importance-sampling correction:

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,r_1^i\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0\right] = \nabla\,\mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0, y)].$$

For the second term, condition on the σ-field generated by the first-turn variables $\mathcal{H} := \sigma\big(\{y_0^j, c_0^j, x_1^j\}_{j=1}^N\big)$. Then $\bar{r}_0$ is $\mathcal{H}$-measurable, and given $\mathcal{H}$ the only randomness in the i-th factor is $y_1^i \sim \pi(\cdot \mid x_1^i)$. Hence,

$$\mathbb{E}\left[\bar{r}_0 \cdot \frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0\right] = \mathbb{E}\left[\bar{r}_0 \cdot \mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0, \mathcal{H}\right] \mid x_0\right].$$

By importance sampling,

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0, \mathcal{H}\right] = \mathbb{E}_{y\sim\pi(\cdot|x_0)}[\nabla \log \pi(y \mid x_0)] = \nabla \int \pi(y \mid x_0)\,dy = 0.$$

Therefore the entire second term is zero, and we obtain

$$\mathbb{E}\left[\frac{\pi(y_1^i \mid x_0)}{\pi(y_1^i \mid x_1^i)}\,(r_1^i - \bar{r}_0)\,\nabla \log \pi(y_1^i \mid x_0) \mid x_0\right] = \nabla\,\mathbb{E}_{y\sim\pi(\cdot|x_0)}[r(x_0, y)].$$

Averaging over i yields the claim. □

B.2 Difference between unbiasedness and point-wise signal collapse

Fix a prompt x_0 and a group of N second-round samples $\{y_1^i\}_{i=1}^N$ with rewards $r_1^i := r(x_0, y_1^i) \in [0, 1]$. Consider a generic (possibly importance-corrected) estimator

$$\hat{g} \;=\; \frac{1}{N}\sum_{i=1}^{N} w_i\,A_i\,s_i, \qquad s_i := \nabla \log \pi(y_1^i \mid x_0),$$

where w_i is a weight (e.g. $w_i = \pi(y_1^i \mid x_0)/\pi(y_1^i \mid x_1^i)$) and A_i is an advantage.

Unbiasedness is an in-expectation statement. Under the assumptions in Appendix B, for suitable choices of A_i and w_i we have $\mathbb{E}[\hat{g} \mid x_0] = \nabla J(\pi)$ (e.g. Eq. (10)), which is a statement about the conditional mean.

Signal collapse is a point-wise (distributional) statement. Even if $\mathbb{E}[\hat{g} \mid x_0]$ matches the target gradient, $\hat{g}$ can be identically zero for a nontrivial fraction of groups, yielding no update on those groups.

Proposition B.3 (Deterministic collapse under second-turn mean baselines)**.** *Let the second-turn baseline be a mean of the same group rewards, e.g. the in-sample mean $\bar{r}_1 := \frac{1}{N}\sum_{j=1}^N r_1^j$ or the leave-one-out mean $\bar{r}_{1,-i} := \frac{1}{N-1}\sum_{j\neq i} r_1^j$, and define $A_i^{(1)} := r_1^i - \bar{r}_1$ (or $r_1^i - \bar{r}_{1,-i}$). If the group rewards are constant, i.e. $r_1^1 = \cdots = r_1^N$, then $A_i^{(1)} \equiv 0$ for all i and hence $\hat{g} \equiv 0$, regardless of the weights $\{w_i\}$.*

Proof. If $r_1^1 = \cdots = r_1^N$, then $\bar{r}_1 = r_1^i$ and also $\bar{r}_{1,-i} = r_1^i$ for each i. Thus each advantage equals zero deterministically, so every summand vanishes. □

Proposition B.4 (How often collapse occurs for Bernoulli rewards)**.** *Assume $r_1^i \in \{0, 1\}$ are i.i.d. given x_0 with $\Pr(r_1^i = 1 \mid x_0) = p_1$. Under either in-sample or leave-one-out second-turn mean baselines, the estimator collapses with probability*

$$\Pr(\hat{g} = 0 \mid x_0) \;\geq\; \Pr(r_1^1 = \cdots = r_1^N \mid x_0) \;=\; p_1^N + (1 - p_1)^N.$$

In particular, when $p_1 \to 1$, the probability of a nonzero *update scales as $1 - p_1^N \approx N(1 - p_1)$.*

Near-collapse and variance interpretation. Let $\bar{r}_1 = \frac{1}{N}\sum_i r_1^i$. For the in-sample mean baseline $A_i^{(1)} = r_1^i - \bar{r}_1$,

$$\frac{1}{N}\sum_{i=1}^{N}(A_i^{(1)})^2 \;=\; \frac{1}{N}\sum_{i=1}^{N}(r_1^i - \bar{r}_1)^2$$

is exactly the empirical variance of the group rewards. Thus whenever the second-turn rewards concentrate (e.g. high success-rate distillation where $r_1^i \approx 1$), the advantages are uniformly small and the update magnitude is small. This complements the in-expectation analysis in Appendix B: unbiasedness controls the mean of $\hat{g}$, whereas collapse is governed by the *mass of $\hat{g}$ near zero*, which can be large when rewards saturate.

B.3 Discussion on Alternative Baselines

A natural alternative to the first-turn mean baseline is the *trajectory-level improvement* advantage

$$A_i^{\Delta} \;:=\; R(x_0, y_1^i) - R(x_0, y_0^i), \tag{9}$$

which measures how much the critique-conditioned revision improves a particular trajectory. This choice is unbiased in our estimator because $R(x_0, y_0^i)$ does not depend on the scored action y_1^i.

Variance comparison. To compare variance, it is helpful to separate the effect of the advantage from the score term. Fix a prompt x_0 and consider the conditional variance of the scalar advantage (the same comparison carries through to the gradient in any fixed direction if the score is uniformly bounded). Let

$$R_1 := R(x_0, y_1), \qquad R_0 := R(x_0, y_0),$$

and let $b^{(0)} = \frac{1}{N}\sum_{j=1}^{N} R(x_0, y_0^j)$ be the first-turn mean baseline used in (5). For a single trajectory i, define

$$A^{(0)} := R_1 - b^{(0)}, \qquad A^{\Delta} := R_1 - R_0.$$

Conditioned on x_0, R_1 is independent of the *other* first-turn samples $\{R(x_0, y_0^j)\}_{j\neq i}$, hence independent of $b^{(0)}$ up to a $1/N$ self-term. Approximating this finite-sample effect by ignoring the self-term (or using the leave-one-out baseline), we have $\mathrm{Cov}(R_1, b^{(0)} \mid x_0) \approx 0$ and therefore

$$\mathrm{Var}(A^{(0)} \mid x_0) \;=\; \mathrm{Var}(R_1 \mid x_0) + \mathrm{Var}(b^{(0)} \mid x_0) \;\approx\; \mathrm{Var}(R_1 \mid x_0) + \frac{1}{N}\mathrm{Var}(R_0 \mid x_0), \tag{10}$$

$$\mathrm{Var}(A^{\Delta} \mid x_0) \;=\; \mathrm{Var}(R_1 - R_0 \mid x_0) \;=\; \mathrm{Var}(R_1 \mid x_0) + \mathrm{Var}(R_0 \mid x_0) - 2\mathrm{Cov}(R_1, R_0 \mid x_0). \tag{11}$$

Subtracting (10) from (11) yields

$$\mathrm{Var}(A^{\Delta} \mid x_0) - \mathrm{Var}(A^{(0)} \mid x_0) \;\approx\; \Big(1 - \frac{1}{N}\Big)\mathrm{Var}(R_0 \mid x_0) - 2\mathrm{Cov}(R_1, R_0 \mid x_0). \tag{12}$$

Equation (12) shows that A^{Δ} has *larger* conditional variance than $A^{(0)}$ whenever

$$\mathrm{Cov}(R_1, R_0 \mid x_0) \;\leq\; \frac{1}{2}\Big(1 - \frac{1}{N}\Big)\mathrm{Var}(R_0 \mid x_0). \tag{13}$$

This is the typical regime in our setting. In sparse-reward problems, R_0 is near-deterministically zero under the base policy (so $\mathrm{Var}(R_0 \mid x_0) \approx p_0(x_0)$), while the dependence between first-turn success and post-feedback success is often weak or even *negative*: critiques primarily help when the first attempt fails, making R_1 and R_0 less positively correlated. In particular, if $\mathrm{Cov}(R_1, R_0 \mid x_0) \approx 0$ (or ≤ 0), then

$$\mathrm{Var}(A^{\Delta} \mid x_0) \;\gtrsim\; \mathrm{Var}(A^{(0)} \mid x_0) + \Big(1 - \frac{1}{N}\Big)\mathrm{Var}(R_0 \mid x_0),$$

so the improvement baseline pays an extra variance term of order $\mathrm{Var}(R_0)$, whereas the mean baseline only pays $\mathrm{Var}(R_0)/N$.

Additional downsides beyond variance. The improvement baseline also changes *what* the algorithm emphasizes:

- **It discards many informative second-turn successes.** If both attempts succeed ($R_0 = R_1 = 1$), then $A^{\Delta} = 0$ and the trajectory contributes no learning signal, even though y_1 may still contain useful "clean" solutions worth distilling into the one-shot policy. In contrast, $A^{(0)} = 1 - b^{(0)}$ remains positive whenever the first-turn policy is imperfect ($b^{(0)} < 1$), so it continues to reinforce successful corrected outputs.
- **It provides weak normalization across prompts.** When $R_0 \approx 0$ for most samples, $A^{\Delta} \approx R_1$ and the method effectively reduces to using raw post-feedback rewards, losing the prompt-level normalization that makes (5) stable when post-feedback success becomes high.

B.4 Recovering Rejection Sampling

Our framework recovers the commonly used Rejection Sampling (or SFT) baseline in distillation: namely, we collect correct second-round generations and then SFT them into the single-turn policy $\pi(\cdot \mid x_0)$. Concretely, for each x_0 we sample $y_0 \sim \pi(\cdot \mid x_0)$, form $x_1 = f(x_0, y_0, c_0)$, and sample $y_1 \sim \pi(\cdot \mid x_1)$. We keep y_1 if it is correct, and perform maximum likelihood on the accepted set.

This procedure fits into Eq. (3) by taking a binary advantage $A(y_1) = R(x_0, y_1) \in \{0, 1\}$, and choosing the reference as the same single-turn policy, but treated as a constant via stop-gradient, i.e., $\pi_{\text{ref}}(\cdot \mid x_0) = \text{sg}[\pi(\cdot \mid x_0)]$. With this choice, the ratio $\pi(y_1 \mid x_0)/\pi_{\text{ref}}(y_1 \mid x_0)$ evaluates to 1 in the forward pass while still producing the desired score-function gradient. In particular, the induced update direction becomes

$$\nabla \ell^{\text{SFT}}_{\textsf{distill}}(\pi) \;:=\; \mathbb{E}_{y_1 \sim \pi(\cdot \mid x_1)} \left[\frac{\pi(y_1 \mid x_0)}{\text{sg}(\pi(y_1 \mid x_0))} \, R(x_0, y_1) \, \nabla \log \pi(y_1 \mid x_0) \right] = \mathbb{E}_{y_1 \sim \pi(\cdot \mid x_1)} [R(x_0, y_1) \, \nabla \log \pi(y_1 \mid x_0)].$$

Therefore, in our setting SFT is precisely the special case that *distills only the correct second-round generations* (via rejection sampling on the 0–1 reward), without any importance-weight variance from off-policy correction.

In practice, we observe that Rejection Sampling underperforms methods with baselines, as we show in Section 5.2.

C THEORY RESULTS FROM SECTION 4

RLTF-FM trains the model to *predict feedback*, not to explicitly output a corrected answer, so its benefit is not obvious a priori. We provide an early-stage analysis through the lens of *representation learning* in a frozen-rollout regime (a batch RL setting, where data are effectively drawn from a fixed distribution d_0) with log-linear (i.e., softmax) policy with learned representation. The central question is: *what representation directions are statistically identifiable from the available training signal under base rollouts?* With the batch setting and log-linear policy, our setting is rather idealized, but the analysis yields useful insights into the benefit of RLTF-FM, which we summarize below.

Reward-only signal can be both *rare* and *geometrically concentrated*. With sparse rewards, especially early in training when the base policy performs poorly, only a small fraction of rollouts succeed. Let ε_0 denote this base pass rate. Then the per-sample policy-gradient estimator has low signal-to-noise ratio, and reliably estimating even a single gradient component can require on the order of $1/\varepsilon_0$ rollouts. Beyond this finite-sample bottleneck, we identify a population-level geometric limitation: even conditioning on success, the reward-only learning signal can concentrate on a small set of representation directions. Equivalently, there can exist a large subspace of directions that are *weakly identified* by reward-only updates under base-policy sampling. In the frozen-rollout regime, we formalize this by defining a *low-signal subspace* S_{low} from success-conditioned score statistics at initialization, and we show that progress in S_{low} along the optimization trajectory is controlled by the cumulative magnitude of the success-conditioned score in that subspace. Intuitively, reward-only RL can therefore behave like an effectively low-rank update under base rollouts, making some task-relevant representation directions difficult to learn without auxiliary supervision.

Feedback modeling supplies a better-conditioned representation signal. In contrast, natural-language feedback is dense and structured. We show that under the same batch regime, RLTF-FM induces nontrivial movement of the shared representation under a *coverage* assumtion, which is analogous to standard coverage in linear/low-rank MDPs (Jin et al., 2021; Uehara & Sun, 2021) or LLM preference learining (Chang et al., 2024; Song et al., 2024a). This matches the high-level intuition that *reward-only RL may provide a narrow (often nearly rank-1) representation signal under base rollouts, whereas feedback modeling yields a better-conditioned information source that can "fill in" missing representation directions.*

We summarize the main results informally below.

Proposition C.1 (Reward-only bottlenecks under base rollouts (informal))**.** *Under the batch regime, and rewards are sparse with base success rate ε_0. Then reward-only learning faces two bottlenecks:*

(i) Rare-event estimation. *Because reward is supported on a low-probability success event, the directional policy-gradient estimator has low signal-to-noise ratio: for any direction,* SNR *scales at most as $\sqrt{\varepsilon_0}$. Consequently, reliably estimating even a single gradient component requires on the order of $1/\varepsilon_0$ rollouts.*

(ii) Weak identifiability of representation directions. *Even conditioning on success, the reward-weighted gradient signal concentrates on a small set of directions in representation space. Equivalently, there can exist a nontrivial* low-signal subspace *of directions that are weakly identified by reward-only updates under base rollouts. In the frozen-rollout regime, reward-only updates can have negligible projection onto these directions over many steps.*

Proposition C.2 (Feedback modeling yields a well-conditioned representation signal (informal))**.** *In the same early-stage frozen-rollout regime, feedback modeling (RLTF-FM) provides an additional supervised learning signal on the shared representation. Under mild conditions on the feedback coverage (Assumption C.3), RLTF-FM is informative in representation directions that are weakly identified by sparse reward under base rollouts. As a result, RLTF-FM can learn representation degrees of freedom that reward-only RL fails to identify early on.*

The analysis explains why RLTF-FM helps without explicitly teaching revision: predicting critiques supplies an additional supervised signal that is informative in representation directions that are weakly identified by sparse reward under base rollouts. In the batch regime, this feedback signal can induce substantial representation learning in a low-signal subspace. From this perspective, RLTF-FM acts like a *representation preconditioner*: it improves the identifiability and conditioning of the representation degrees of freedom that reward-only RL struggles to learn early on. Beyond offering insights into

RLTF-FM, our theory provides a framework and techniques with standalone merits and broader applicability.

C.1 Setup

We analyze an early-stage (batch) regime of RL post-training through a horizon-1 contextual bandit abstraction. A prompt (context) is sampled as $x \sim \mu$, the model outputs a response $y \in \mathcal{Y}(x)$, and receives a bounded reward $R(x, y) \in [0, 1]$. The objective is

$$J(\pi) \;:=\; \mathbb{E}_{x\sim\mu}\,\mathbb{E}_{y\sim\pi(\cdot|x)}[R(x, y)].$$

Log-linear policy with learned representation. We parameterize the policy by a learned representation $z_w(x, y) \in \mathbb{R}^d$ and a linear head $b \in \mathbb{R}^d$, with parameters $\theta = (b, w)$. Define the score function

$$f_\theta(x, y) \;:=\; b^\top z_w(x, y),$$

and the log-linear policy

$$\pi_\theta(y \mid x) \;:=\; \frac{\exp(\tau f_\theta(x, y))}{\sum_{y'\in\mathcal{Y}(x)} \exp(\tau f_\theta(x, y'))}, \tag{14}$$

where $\tau > 0$ is an inverse temperature.

Frozen rollout distribution. We study an early-stage regime in which rollouts remain close to the base policy, so samples are effectively drawn from a fixed distribution

$$d_0(x, y) \;:=\; \mu(x)\,\pi_{\theta_0}(y \mid x), \qquad \theta_0 = (b_0, w_0).$$

Equivalently, we take $\pi_{\mathsf{base}} := \pi_{\theta_0}$. This frozen-rollout assumption is the standing regime for the directional SNR calculations in Section C.1 and the trajectory bounds in Section C.2.

REINFORCE estimator and score features. Let

$$s_\theta(x, y) \;:=\; \nabla_\theta \log \pi_\theta(y \mid x), \qquad g(x, y) \;:=\; s_{\theta_0}(x, y)$$

denote the score at initialization. The reward-only REINFORCE estimator at θ_0 is

$$\widehat{g}(x, y) \;:=\; R(x, y)\, g(x, y), \qquad (x, y) \sim d_0,$$

and the corresponding population gradient is

$$g_{\mathrm{RL}}(\theta_0) \;:=\; \nabla_\theta J(\pi_\theta)\big|_{\theta=\theta_0} \;=\; \mathbb{E}_{(x,y)\sim d_0}[R(x, y)\, g(x, y)]\,.$$

Fisher information and reward-weighted second moment. Define the (rollout) Fisher information matrix at θ_0 by

$$I(\theta_0) \;:=\; \mathbb{E}_{d_0}\big[g(x, y)g(x, y)^\top\big]\,.$$

For the linear head b, letting

$$\phi_{\theta_0}(x, y) := z_{w_0}(x, y) - \mathbb{E}_{y'\sim\pi_{\theta_0}(\cdot|x)}[z_{w_0}(x, y')],$$

we have the closed-form score

$$\nabla_b \log \pi_{\theta_0}(y \mid x) = \tau\, \phi_{\theta_0}(x, y),$$

so the Fisher restricted to b equals the policy-induced feature covariance:

$$I_b(\theta_0) \;=\; \tau^2\, \mathbb{E}_{x\sim\mu}\Big[\mathrm{Cov}_{y\sim\pi_{\theta_0}(\cdot|x)}\big(z_{w_0}(x, y)\big)\Big]\,.$$

We will also use the reward-weighted score second moment

$$\Sigma_{\mathrm{RL}}(\theta_0) \;:=\; \mathbb{E}_{(x,y)\sim d_0}\big[R(x, y)^2\, g(x, y)g(x, y)^\top\big]\,. \tag{15}$$

For any unit direction u, the second moment of the directional estimator equals

$$M_{2,u} \;:=\; \mathbb{E}\big[\langle \widehat{g}, u\rangle^2\big] \;=\; u^\top \Sigma_{\mathrm{RL}}(\theta_0)\, u.$$

Directional SNR. Fix any unit direction u and define

$$Z_u \;:=\; \langle \widehat{g}, u\rangle \;=\; R(x,y)\,\langle g(x,y), u\rangle.$$

Let $\mu_u := \mathbb{E}[Z_u]$ and $M_{2,u} := \mathbb{E}[Z_u^2]$ under $(x,y) \sim d_0$. The per-sample directional signal-to-noise ratio is

$$\mathrm{SNR}(u) \;:=\; \frac{|\mu_u|}{\sqrt{M_{2,u}}}. \tag{16}$$

This quantity controls the sample complexity required to reliably estimate a gradient component along direction u.

Lemma C.1 (Directional concentration in terms of SNR)**.** *Let $Z_{u,1}, \ldots, Z_{u,N}$ be i.i.d. samples of Z_u and $\overline{Z}_u := \frac{1}{N}\sum_{i=1}^{N} Z_{u,i}$. Assume $R \in [0,1]$ and $|\langle g(x,y), u\rangle| \le G_u$ almost surely under d_0, so $|Z_u| \le G_u$. Then for any $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$|\overline{Z}_u - \mu_u| \;\le\; \sqrt{\frac{2\,M_{2,u}\,\log(2/\delta)}{N}} \;+\; \frac{4G_u \log(2/\delta)}{3N}. \tag{17}$$

Consequently, for any constant $\alpha \in (0,1)$, obtaining $|\overline{Z}_u - \mu_u| \le \alpha|\mu_u|$ (and hence recovering the sign of μ_u when $\alpha < 1$) requires

$$N \;=\; \Omega\!\left(\frac{1}{\mathrm{SNR}(u)^2}\log\frac{1}{\delta}\right),$$

up to constant factors and lower-order $1/N$ terms.

C.2 Reward-only RL under base rollouts

We present two complementary bottlenecks for reward-only policy gradients under the frozen rollout distribution $d_0(x,y) = \mu(x)\pi_{\theta_0}(y \mid x)$. Both bottlenecks are stated in terms of the directional statistics in Section C.1: $Z_u = R\langle g, u\rangle$, $\mu_u = \mathbb{E}[Z_u]$, $M_{2,u} = \mathbb{E}[Z_u^2]$, and $\mathrm{SNR}(u) = |\mu_u|/\sqrt{M_{2,u}}$.

C.2.1 Rare-event regime: small success probability implies low directional SNR

We first formalize the common regime where reward is supported on a low-probability success event.

Lemma C.2 (SNR bound under reward supported on a rare event)**.** *Let $S(x,y) \in \{0,1\}$ be any event and define $\varepsilon_0 := \Pr_{(x,y)\sim d_0}(S=1)$. Assume the reward is supported on S, i.e., $R(x,y) = 0$ whenever $S(x,y) = 0$. Then for any unit direction u,*

$$\mathrm{SNR}(u) = \sqrt{\varepsilon_0}\cdot\frac{\big|\mathbb{E}[R(x,y)\langle g(x,y),u\rangle \mid S=1]\big|}{\sqrt{\mathbb{E}[R(x,y)^2\langle g(x,y),u\rangle^2 \mid S=1]}} \;\le\; \sqrt{\varepsilon_0}.$$

Corollary C.1 (Sample complexity under small pass rate)**.** Assume binary reward $R(x,y) = \mathbf{1}\{\mathrm{pass}(x,y)\}$ and let $S(x,y) = \mathbf{1}\{\mathrm{pass}(x,y)\}$. Then $\varepsilon_0 = \Pr_{(x,y)\sim d_0}(\mathrm{pass}=1)$ is the base pass rate and the assumption of Lemma C.2 holds. Hence for all unit u, $\mathrm{SNR}(u) \le \sqrt{\varepsilon_0}$. Consequently, with probability at least $1-\delta$, recovering $\mathrm{sign}(\mu_u)$ for any unit direction u requires

$$N \;=\; \Omega\!\left(\frac{1}{\varepsilon_0}\log\frac{1}{\delta}\right)$$

rollouts.

C.2.2 Weak identifiability of representation directions under success conditioning

Motivation. Corollary C.1 is a finite-sample statement: when successes are rare, large batches are required to reliably estimate gradient components. We now isolate a population-level geometric limitation that can persist even with access to the exact population gradient under d_0, focusing on the representation parameters w with $b = b_0$ fixed.

Assumption C.1 (Frozen rollout distribution for the first T steps)**.** *For the first T gradient steps, all expectations defining the update are taken under the same fixed distribution $(x, y) \sim d_0(x, y) = \mu(x)\pi_{\theta_0}(y \mid x)$, and $R(x, y) = \mathbf{1}\{\mathrm{pass}(x, y)\}$ is evaluated on samples from d_0.*

Define the representation score (with head fixed at $b = b_0$)

$$s_w^w(x, y) := \nabla_w \log \pi_{(b_0,w)}(y \mid x) \in \mathbb{R}^p, \qquad \varepsilon_0 := \Pr_{(x,y)\sim d_0}(\mathrm{pass}(x, y) = 1),$$

and the success-conditioned representation score second moment under d_0

$$\Sigma_{\mathrm{succ}}^w(w) := \mathbb{E}\big[s_w^w(x, y)\, s_w^w(x, y)^\top \;\big|\; \mathrm{pass}(x, y) = 1\big] \in \mathbb{R}^{p\times p}.$$

Let $\Sigma_{\mathrm{succ}}^w(w_0) = \sum_{i=1}^m \lambda_i v_i v_i^\top$ with $\lambda_1 \geq \cdots \geq \lambda_m \geq 0$ and define the low-signal subspace and projector

$$S_{\mathrm{low}}(\eta) := \mathrm{span}\{v_i : \lambda_i < \eta\}, \qquad \Pi := \Pi_{S_{\mathrm{low}}(\eta)}.$$

Interpretation. The spectrum of $\Sigma_{\mathrm{succ}}^w(w_0)$ quantifies which representation directions are statistically identifiable from success-conditioned policy scores under base rollouts. Small eigenvalues indicate directions in which even successful samples carry little score second moment, so reward-weighted updates have negligible projection onto those directions at initialization under d_0.

Lemma C.3 (Projected gradient bound under fixed d_0)**.** *Adopt Assumption C.1 and consider the fixed-d_0 reward gradient field*

$$g_{\mathrm{RL}}^w(w) := \mathbb{E}_{(x,y)\sim d_0}[R(x, y)\, s_w^w(x, y)]\,.$$

Then for any w,

$$\|\Pi g_{\mathrm{RL}}^w(w)\|_2 \;\leq\; \varepsilon_0 \sqrt{\|\Pi\, \Sigma_{succ}^w(w)\, \Pi\|_{\mathrm{op}}}\,.$$

In particular, $\|\Pi g_{\mathrm{RL}}^w(w_0)\|_2 \leq \varepsilon_0\sqrt{\eta}$.

Theorem C.1 (Low-signal progress bound under fixed d_0)**.** *Adopt Assumption C.1 and consider exact gradient ascent updates under fixed d_0:*

$$w_{t+1} = w_t + \rho\, g_{\mathrm{RL}}^w(w_t), \qquad g_{\mathrm{RL}}^w(w) := \mathbb{E}_{(x,y)\sim d_0}[R(x, y)\, s_w^w(x, y)]\,.$$

Then for any integer $T \geq 1$,

$$\|\Pi(w_T - w_0)\|_2 \;\leq\; \rho\, \varepsilon_0 \sum_{t=0}^{T-1} \sqrt{\|\Pi\, \Sigma_{succ}^w(w_t)\, \Pi\|_{\mathrm{op}}}\,.$$

Remark C.1. Theorem C.1 is a statement about the representation parameters w with the head held fixed at $b = b_0$. It does not rule out the possibility that optimization may also be limited by the linear head b (or by joint (b, w) interactions) in other regimes. Our purpose here is narrower: we isolate representation directions that are weakly identified by reward-only learning under base rollouts, to compare with the representation movement induced by RLTF-FM in Section C.3.

Remark C.2 (Interpretation)**.** Under frozen base rollouts $(x, y) \sim d_0$, projected progress in the low-signal subspace $S_{\mathrm{low}}(\eta)$ is controlled by the cumulative success-conditioned score second moment along the trajectory,

$$\sum_{i<t} \sqrt{\|\Pi\Sigma_{\mathrm{succ}}^w(w_i)\Pi\|_{\mathrm{op}}}.$$

Thus, even with a fixed data distribution, an early-stage plateau can occur when successful samples have small success-conditioned score second moment in $S_{\mathrm{low}}(\eta)$ over the window of interest.

Corollary C.2 (An illustrative plateau condition)**.** Adopt the assumptions of Theorem C.1. Fix any $\Delta > 0$ and define the superlevel set

$$\Theta_\Delta \;:=\; \big\{w \;:\; J(\pi_{(b_0,w)}) \;\geq\; J(\pi_{(b_0,w_0)}) + \Delta\big\}\,.$$

Define the required displacement in the low-signal subspace by

$$r_\Delta \;:=\; \inf_{w\in\Theta_\Delta} \|\Pi(w - w_0)\|_2.$$

Assume $r_\Delta > 0$ and define the trajectory-dependent cumulative score second moment quantity

$$\mathcal{E}_t \;:=\; \sum_{i=0}^{t-1} \sqrt{\|\Pi\, \Sigma^w_{\text{succ}}(w_i)\, \Pi\|_{\text{op}}}\,.$$

Then for any $t \le T$, if $\rho\, \varepsilon_0\, \mathcal{E}_t < r_\Delta$, the iterates satisfy

$$J(\pi_{(b_0,w_t)}) \;<\; J(\pi_{(b_0,w_0)}) + \Delta.$$

Equivalently, achieving $J(\pi_{(b_0,w_t)}) \ge J(\pi_{(b_0,w_0)}) + \Delta$ within this regime requires $\rho\, \varepsilon_0\, \mathcal{E}_t \ge r_\Delta$.

Remark C.3. The condition $r_\Delta > 0$ is problem-dependent: it asserts that achieving a Δ improvement in expected reward with fixed head b_0 requires nontrivial movement in the low-signal subspace as measured by $\|\Pi(w - w_0)\|_2$. We include Corollary C.2 to make explicit how a low-signal representation subspace can induce an early-stage plateau under base rollouts, in terms of the cumulative success-conditioned score second moment $\mathcal{E}_t$ supplied by successful samples.

C.3 Feedback modeling: representation learning benefit in the RL low-signal subspace

We isolate a representation-learning benefit of RLTF-FM in the same early-stage frozen-rollout regime as Section C.2. We do not prove reward improvement here. Instead, we formalize the following claim:

> *Under base rollouts, reward-only learning can have negligible driving signal in a low-signal representation subspace $S_{\text{low}}(\eta)$ defined in Section C.2. In contrast, RLTF-FM provides auxiliary supervision that induces nontrivial representation movement in this subspace, so these degrees of freedom become statistically identifiable even before the rollout distribution shifts.*

C.3.1 Setup

We model a shared representation $z_w(x, y) \in \mathbb{R}^d$ with $w \in \mathbb{R}^p$. Write its Jacobian at initialization as

$$J(x, y) \;:=\; \nabla_w z_w(x, y)\big|_{w=w_0} \in \mathbb{R}^{d\times p},$$

so that $J(x, y)^\top v \in \mathbb{R}^p$ for any $v \in \mathbb{R}^d$.

Policy score in representation space. Holding the head fixed at $b = b_0$, the policy representation score for Eq. (14) is

$$s^w_w(x, y) = \nabla_w \log \pi_{(b_0,w)}(y \mid x) = \tau\Big(J(x, y)^\top b_0 - \mathbb{E}_{y'\sim\pi_{(b_0,w)}(\cdot|x)}[J(x, y')^\top b_0]\Big).$$

In particular, at w_0 this matches the score used to define $\Sigma^w_{\text{succ}}(w_0)$ and $S_{\text{low}}(\eta)$ in Section C.2.

Feedback model. Let $\mathcal{C}$ be the set of critique token/types and let $u_\psi(c) \in \mathbb{R}^d$ be a class embedding for critique type c, parameterized by ψ. Define the feedback model

$$p_{\psi,w}(c \mid x, y) \;:=\; \frac{\exp\big(\langle u_\psi(c), z_w(x, y)\rangle\big)}{\sum_{c'\in\mathcal{C}} \exp\big(\langle u_\psi(c'), z_w(x, y)\rangle\big)}.$$

Its representation score at (ψ_0, w_0) is

$$s_{\text{FM}}(x, y, c) := \nabla_w \log p_{\psi_0,w}(c \mid x, y)\big|_{w=w_0} = J(x, y)^\top\Big(u_{\psi_0}(c) - \mathbb{E}_{c'\sim p_{\psi_0,w_0}(\cdot|x,y)}[u_{\psi_0}(c')]\Big) \in \mathbb{R}^p.$$

All expectations below are under $(x, y) \sim d_0 := \mu(x)\pi_{\theta_0}(y \mid x)$ and $c \sim \mathcal{M}(\cdot \mid x, y)$.

Recall from Section C.2 that $S_{\text{low}}(\eta) \subseteq \mathbb{R}^p$ and $\Pi \in \mathbb{R}^{p\times p}$ are defined via the success-conditioned second moment $\Sigma^w_{\text{succ}}(w_0)$ of the policy representation score.

C.3.2 ASSUMPTIONS

Define the FM score mean and centered covariance at (ψ_0, w_0):

$$m_{\mathrm{FM}} := \mathbb{E}[s_{\mathrm{FM}}(x, y, c)], \qquad C_{\mathrm{FM}} := \mathbb{E}\big[(s_{\mathrm{FM}}(x, y, c) - m_{\mathrm{FM}})(s_{\mathrm{FM}}(x, y, c) - m_{\mathrm{FM}})^\top\big] .$$

Assumption C.2 (FM drift in the RL low-signal subspace)**.** *There exists* $b_{\mathrm{FM}} > 0$ *such that* $\|\Pi m_{\mathrm{FM}}\|_2 \geq b_{\mathrm{FM}}$.

A sufficient interpretation is that, under base rollouts, the feedback model p_{ψ_0,w_0} is moment-mismatched with the feeder $\mathcal{M}$ along directions in $S_{\mathrm{low}}(\eta)$. Indeed,

$$m_{\mathrm{FM}} = \mathbb{E}_{(x,y)\sim d_0}\Big[J(x, y)^\top \Big(\mathbb{E}_{c\sim\mathcal{M}(\cdot|x,y)}[u_{\psi_0}(c)] - \mathbb{E}_{c\sim p_{\psi_0,w_0}(\cdot|x,y)}[u_{\psi_0}(c)]\Big)\Big].$$

Assumption C.3 (FM coverage)**.** *There exists* $\gamma_{\mathrm{FM}} > 0$ *such that*

$$\Pi\, C_{\mathrm{FM}}\, \Pi \succeq \gamma_{\mathrm{FM}} \Pi.$$

This assumption is a covariance conditioning requirement: the feedback score covariance is non-degenerate on $S_{\mathrm{low}}(\eta)$ under base rollouts, so directions in this subspace are statistically identifiable from feedback supervision.

C.3.3 RESULT: REPRESENTATION-LEARNING BENEFIT

We analyze FM-only updates on the shared representation parameters using the frozen score evaluated at w_0:

$$w_{t+1} = w_t + \rho\lambda\, s_{\mathrm{FM}}(x_t, y_t, c_t)\big|_{w=w_0}, \qquad (x_t, y_t) \sim d_0, \;\; c_t \sim \mathcal{M}(\cdot \mid x_t, y_t),$$

with step size $\rho > 0$ and FM weight $\lambda > 0$.

Theorem C.2 (FM moves the shared representation in the RL low-signal subspace)**.** *Let* $k := \mathrm{tr}(\Pi) = \dim(S_{\mathrm{low}}(\eta))$. *For any integer* $T \geq 1$,

$$\mathbb{E}\big[\|\Pi(w_T - w_0)\|_2^2\big] = \rho^2\lambda^2\Big(T\,\mathrm{tr}(\Pi C_{\mathrm{FM}}\Pi) \;+\; T^2\,\|\Pi m_{\mathrm{FM}}\|_2^2\Big).$$

In particular, under Assumptions C.2 and C.3,

$$\mathbb{E}\big[\|\Pi(w_T - w_0)\|_2^2\big] \;\geq\; \rho^2\lambda^2\Big(T\,\gamma_{\mathrm{FM}}\,k \;+\; T^2\,b_{\mathrm{FM}}^2\Big).$$

Remark C.4. Theorem C.2 decomposes movement in $S_{\mathrm{low}}(\eta)$ into a covariance term $T\,\mathrm{tr}(\Pi C_{\mathrm{FM}}\Pi)$ (coverage) and a mean-drift term $T^2\|\Pi m_{\mathrm{FM}}\|_2^2$ (moment mismatch). For large T, the drift term dominates whenever $\|\Pi m_{\mathrm{FM}}\|_2$ is not extremely small, so systematic feeder/model moment mismatch can be the primary driver of representation movement in this regime. Coverage becomes most important when drift is weak (e.g., after partial fitting of the feedback model) or when one seeks broadly conditioned updates across $S_{\mathrm{low}}(\eta)$ rather than movement concentrated along a single biased direction.

Remark C.5. Theorem C.2 is a frozen-score (linearized) calculation: it characterizes the initial FM signal under samples from d_0 by evaluating the score at w_0. In contrast, the reward-only analysis in Theorem C.1 is stated along the optimization trajectory w_t (under the same frozen rollout distribution d_0), and it upper bounds projected progress by a path integral of success-conditioned score second moment. Thus, these results serve different roles rather than forming symmetric global convergence statements: Theorem C.2 establishes that the FM vector field has nontrivial components in $S_{\mathrm{low}}(\eta)$ at initialization (signal availability), whereas Theorem C.1 shows that reward-only progress in that subspace is limited unless successful samples carry substantial success-conditioned score second moment in those directions along the trajectory. Extending Theorem C.2 to a trajectory-level FM analysis would require controlling how the FM score statistics evolve as w changes.

Remark C.6 (Representation-learning benefit under base rollouts)**.** Section C.2 identifies $S_{\mathrm{low}}(\eta)$ (defined from $\Sigma_{\mathrm{succ}}^{w}(w_0)$) as representation directions that can be weakly identified by reward-only learning under base rollouts. Theorem C.2 shows that, in the same frozen distribution regime, RLTF-FM induces nontrivial representation movement along these directions through a systematic mean component Πm_{FM} and covariance conditioning $\Pi C_{\mathrm{FM}}\Pi$. In this sense, RLTF-FM can make low-signal representation degrees of freedom statistically identifiable from feedback supervision before any reward-improvement guarantee is invoked.

C.4 PROOFS

Proof of Lemma C.1. Let $X_i := Z_{u,i} - \mu_u$. Then $\mathbb{E}[X_i] = 0$ and

$$|X_i| \le |Z_{u,i}| + |\mu_u| \le G_u + \mathbb{E}|Z_u| \le 2G_u \qquad \text{a.s.}$$

Moreover, $\mathrm{Var}(X_i) = \mathrm{Var}(Z_u) \le \mathbb{E}[Z_u^2] = M_{2,u}$. By Bernstein's inequality for bounded i.i.d. mean-zero variables, for any $\delta \in (0,1)$, with probability at least $1-\delta$,

$$\left|\frac{1}{N}\sum_{i=1}^{N} X_i\right| \le \sqrt{\frac{2\,\mathrm{Var}(Z_u)\,\log(2/\delta)}{N}} + \frac{2(2G_u)\log(2/\delta)}{3N}.$$

Since $\frac{1}{N}\sum_{i=1}^{N} X_i = \overline{Z}_u - \mu_u$, and $\mathrm{Var}(Z_u) \le M_{2,u}$, this yields (17).

For the relative-error condition, it suffices that each term on the right-hand side of (17) is at most $\frac{\alpha}{2}|\mu_u|$. The square-root term condition gives

$$\sqrt{\frac{2M_{2,u}\log(2/\delta)}{N}} \le \frac{\alpha}{2}|\mu_u| \quad \Longleftrightarrow \quad N \ge \frac{8M_{2,u}}{\alpha^2\mu_u^2}\log\frac{2}{\delta}.$$

The linear term condition gives

$$\frac{4G_u\log(2/\delta)}{3N} \le \frac{\alpha}{2}|\mu_u| \quad \Longleftrightarrow \quad N \ge \frac{8G_u}{3\alpha|\mu_u|}\log\frac{2}{\delta}.$$

Combining the two yields and substituting $\mathrm{SNR}(u) = |\mu_u|/\sqrt{M_{2,u}}$ gives the stated $\Omega(1/\mathrm{SNR}(u)^2)$ dependence. □

Proof of Lemma C.2. Since $R(x,y) = 0$ on $\{S=0\}$, we have $Z_u = 0$ on $\{S=0\}$ as well, and hence

$$\mu_u = \mathbb{E}[Z_u] = \mathbb{E}[Z_u \mathbf{1}\{S=1\}] = \Pr(S=1)\cdot\mathbb{E}[Z_u \mid S=1] = \varepsilon_0\,\mathbb{E}[Z_u \mid S=1].$$

Similarly,

$$M_{2,u} = \mathbb{E}[Z_u^2] = \mathbb{E}[Z_u^2\mathbf{1}\{S=1\}] = \varepsilon_0\,\mathbb{E}[Z_u^2 \mid S=1].$$

Therefore,

$$\mathrm{SNR}(u) = \frac{|\mu_u|}{\sqrt{M_{2,u}}} = \sqrt{\varepsilon_0}\cdot\frac{|\mathbb{E}[Z_u \mid S=1]|}{\sqrt{\mathbb{E}[Z_u^2 \mid S=1]}}.$$

By Jensen, $|\mathbb{E}[Z_u \mid S=1]| \le \sqrt{\mathbb{E}[Z_u^2 \mid S=1]}$, so the ratio is at most 1. Substituting $Z_u = R(x,y)\langle g(x,y), u\rangle$ yields the displayed expression and the bound $\mathrm{SNR}(u) \le \sqrt{\varepsilon_0}$. □

Proof of Corollary C.1. The first claim follows immediately from Lemma C.2 by taking $S = \mathbf{1}\{\text{pass}\}$ and noting that binary R is supported on S.

For the sample complexity statement, apply Lemma C.1 to the same scalar Z_u with any constant $\alpha \in (0,1)$ (e.g., $\alpha = \frac{1}{2}$). Lemma C.1 states that achieving $|\overline{Z}_u - \mu_u| \le \alpha|\mu_u|$ with probability at least $1-\delta$ requires

$$N = \Omega\left(\frac{1}{\mathrm{SNR}(u)^2}\log\frac{1}{\delta}\right) \quad \text{(up to constant factors and lower-order terms).}$$

Since $\mathrm{SNR}(u) \le \sqrt{\varepsilon_0}$, we have $1/\mathrm{SNR}(u)^2 \ge 1/\varepsilon_0$, yielding

$$N = \Omega\left(\frac{1}{\varepsilon_0}\log\frac{1}{\delta}\right).$$

Finally, when $\alpha < 1$, the condition $|\overline{Z}_u - \mu_u| \le \alpha|\mu_u|$ implies $\overline{Z}_u$ has the same sign as μ_u, so $\mathrm{sign}(\mu_u)$ is recovered. □

Proof of Lemma C.3. Fix any unit $u \in \mathrm{range}(\Pi)$. Since $R = \mathbf{1}\{\text{pass}\}$,

$$\langle g_{\mathrm{RL}}^w(w), u\rangle = \mathbb{E}_{d_0}\big[R\,\langle s_w^w, u\rangle\big] = \varepsilon_0\cdot\mathbb{E}\big[\langle s_w^w(x,y), u\rangle \mid \text{pass}=1\big].$$

By Cauchy–Schwarz,

$$|\langle g^w_{\mathrm{RL}}(w), u\rangle| \le \varepsilon_0 \sqrt{\mathbb{E}\big[\langle s^w_w(x,y), u\rangle^2 \mid \mathrm{pass}=1\big]} = \varepsilon_0 \sqrt{u^\top \Sigma^w_{\mathrm{succ}}(w) u}.$$

Taking the supremum over unit $u \in \mathrm{range}(\Pi)$ yields

$$\|\Pi g^w_{\mathrm{RL}}(w)\|_2 = \sup_{\substack{u \in \mathrm{range}(\Pi) \\ \|u\|_2 = 1}} \langle g^w_{\mathrm{RL}}(w), u\rangle \le \varepsilon_0 \sqrt{\|\Pi \Sigma^w_{\mathrm{succ}}(w) \Pi\|_{\mathrm{op}}}\,.$$

□

Proof of Theorem C.1. By telescoping,

$$\Pi(w_T - w_0) = \rho \sum_{t=0}^{T-1} \Pi g^w_{\mathrm{RL}}(w_t),$$

so by triangle inequality,

$$\|\Pi(w_T - w_0)\|_2 \le \rho \sum_{t=0}^{T-1} \|\Pi g^w_{\mathrm{RL}}(w_t)\|_2.$$

Apply Lemma C.3 term-by-term concludes the proof. □

Proof of Corollary C.2. By Theorem C.1, for any $t \le T$,

$$\|\Pi(w_t - w_0)\|_2 \;\le\; \rho\, \varepsilon_0 \sum_{i=0}^{t-1} \sqrt{\|\Pi\, \Sigma^w_{\mathrm{succ}}(w_i)\, \Pi\|_{\mathrm{op}}} \;=\; \rho\, \varepsilon_0\, \mathcal{E}_t.$$

Thus, if $\rho\, \varepsilon_0\, \mathcal{E}_t < r_\Delta$, then $\|\Pi(w_t - w_0)\|_2 < r_\Delta$.

By definition of $r_\Delta = \inf_{w \in \Theta_\Delta} \|\Pi(w - w_0)\|_2$, every $w \in \Theta_\Delta$ satisfies $\|\Pi(w - w_0)\|_2 \ge r_\Delta$. Therefore $\|\Pi(w_t - w_0)\|_2 < r_\Delta$ implies $w_t \notin \Theta_\Delta$, i.e.,

$$J(\pi_{(b_0, w_t)}) \;<\; J(\pi_{(b_0, w_0)}) + \Delta.$$

The final statement is just the contrapositive: if $J(\pi_{(b_0,w_t)}) \ge J(\pi_{(b_0,w_0)}) + \Delta$ then $w_t \in \Theta_\Delta$, so $\|\Pi(w_t - w_0)\|_2 \ge r_\Delta$, hence by the theorem $\rho\, \varepsilon_0\, \mathcal{E}_t \ge r_\Delta$. □

Proof of Theorem C.2. Let

$$s_t \;:=\; s_{\mathrm{FM}}(x_t, y_t, c_t)\big|_{w=w_0},$$

where $(x_t, y_t) \sim d_0$ and $c_t \sim \mathsf{Feeder}(\cdot \mid x_t, y_t)$ are i.i.d. across t. Then $\{s_t\}_{t=1}^T$ are i.i.d. with mean m_{FM} and centered covariance C_{FM}.

By the update rule,

$$w_T - w_0 \;=\; \rho\lambda \sum_{t=1}^T s_t, \qquad \text{so} \qquad \Pi(w_T - w_0) \;=\; \rho\lambda \sum_{t=1}^T \Pi s_t.$$

Expand the squared norm:

$$\|\Pi(w_T - w_0)\|_2^2 = \rho^2\lambda^2 \Big\| \sum_{t=1}^T \Pi s_t \Big\|_2^2 = \rho^2\lambda^2 \sum_{i=1}^T \sum_{j=1}^T \langle \Pi s_i, \Pi s_j\rangle.$$

Take expectation and use independence. For $i \ne j$,

$$\mathbb{E}\langle \Pi s_i, \Pi s_j\rangle = \big\langle \mathbb{E}[\Pi s_i],\, \mathbb{E}[\Pi s_j]\big\rangle = \|\Pi m_{\mathrm{FM}}\|_2^2.$$

For $i = j$, write $s_i = (s_i - m_{\mathrm{FM}}) + m_{\mathrm{FM}}$ and note

$$\mathbb{E}\|\Pi s_i\|_2^2 = \mathbb{E}\|\Pi(s_i - m_{\mathrm{FM}})\|_2^2 + \|\Pi m_{\mathrm{FM}}\|_2^2 = \mathrm{tr}\big(\Pi C_{\mathrm{FM}} \Pi\big) + \|\Pi m_{\mathrm{FM}}\|_2^2.$$

Therefore,

$$\begin{aligned}\mathbb{E}\big[\|\Pi(w_T - w_0)\|_2^2\big] &= \rho^2\lambda^2\Big(T\big(\mathrm{tr}(\Pi C_{\mathrm{FM}}\Pi) + \|\Pi m_{\mathrm{FM}}\|_2^2\big) + T(T-1)\,\|\Pi m_{\mathrm{FM}}\|_2^2\Big)\\ &= \rho^2\lambda^2\Big(T\,\mathrm{tr}(\Pi C_{\mathrm{FM}}\Pi) + T^2\,\|\Pi m_{\mathrm{FM}}\|_2^2\Big),\end{aligned}$$

which proves the identity.

For the lower bound, Assumption C.3 gives $\Pi C_{\mathrm{FM}}\Pi \succeq \gamma_{\mathrm{FM}}\Pi$, hence

$$\mathrm{tr}(\Pi C_{\mathrm{FM}}\Pi) \;\geq\; \gamma_{\mathrm{FM}}\mathrm{tr}(\Pi) = \gamma_{\mathrm{FM}}\,\dim(S_{\mathrm{low}}(\eta)).$$

Assumption C.2 gives $\|\Pi m_{\mathrm{FM}}\|_2 \;\geq\; b_{\mathrm{FM}}$. Substituting these into the identity yields the stated bound. □

D Experiment Details

D.1 Prompts

Feedback Provider Prompt

You are an expert grader for math/logic problems.
Problem: {PROBLEM}
Student Solution: {LEARNER RESPONSE}
Your task:

- Analyze the student solution step by step.
- Focus on correctness and logical consistency.
- Identify potential mistake(s), if any.
- Provide concrete, actionable hints to improve the solution.
- Keep the Critique section under 200 words.

Format your response exactly as:
Thinking: [Your step-by-step analysis]
Critique: [Your final critique in under 200 words, ending with either "Your previous attempt was correct." or "Your previous attempt was incorrect."]

Policy Prompt

Question: {QUESTION}
You are given your previous attempt and an expert critique of it below. Your task is to produce an improved solution using the critique.
Your Previous Solution: {PREVIOUS RESPONSE}
Expert Critique: {TEXT FEEDBACK}
Instructions:

- Write your answer as a fresh solution to the original problem. Do not refer to your previous attempt.
- Do not mention or refer to the critique or the revision process.
- Use the critique only to improve correctness, clarity, and reasoning.
- Avoid using phrases like "Correctly applying the critique..." or "Reexamining my earlier solution...", etc., as the final answer should stand alone.

Let's think step by step and output the final answer within boxed{}.

D.2 Case Studies

To demonstrate the purpose of text feedback, we study some example generations from the `Shortest Path` task. The goal is to output a sequence of moves (e.g., "right, down, right") from the start (*) to the destination (#), without passing through the obstacles (X). First, we show below an example from earlier in training (step 156), where the first attempt is incorrect and the second attempt is correct. The model incorrectly states that it is infeasible to get to the destination. The expert critique localizes the error by pointing out that moving down to (1,1) is valid. In the second attempt, the model incorporates this feedback and correctly outputs the sequence "right, down, down, down." (Interestingly, we observed that in regular single-turn GRPO, the model gets stuck in the local optimum of outputting "infeasible" regardless of the input; text feedback is critical for escaping this pitfall). Second, we show an example from the end of training (step 616), to show that the model has incorporated improved reasoning skills into even its first attempt capabilities.

Shortest Path Example (Step 123)

Question:
Find the shortest path from the start (*) to the destination (#) in this grid. You can only move through open cells (O) and cannot pass through obstacles (X). Valid moves are: right, down, left, up.

Grid:
X O X X O O
O X O O X X
X # O X O O
O O O O X X
X X O X X O
O X * O O X
If a path exists, return it as a sequence of moves (e.g., "right, down, right"). If no path exists, return "infeasible". Let's think step by step and output the final answer within boxed{}.

First Attempt:
Step 1: Identify the start and destination cells in the grid
The start (*) is located at row 5, column 5, and the destination (#) is at row 2, column 2.
Step 2: Examine the immediate surroundings of the start cell
The start cell at row 5, column 5 is blocked by an obstacle (X) to the north.
Step 3: Investigate possible movements from the start cell
There is no valid movement to the left or up from the start cell due to obstacles (X).
Step 4: Determine the feasibility of reaching the destination
Given that the start cell at row 5, column 5 is blocked by an obstacle to the north, and there are no valid movements to the left or up, it is not possible to reach the destination (#) from the start (*).
Step 5: Conclusion on the path from start to destination
The analysis reveals that the start cell is blocked by an obstacle and there are no valid movements to the left or up, making it infeasible to reach the destination (#) from the start (*).
The final answer is: infeasible

Expert Critique:
Your previous attempt was incorrect. You misidentified the start and destination coordinates. Double-check the grid by counting rows and columns carefully, starting from the top-left as (0,0). The start (*) is at row 5, column 2 (0-indexed), and the destination (#) is at row 2, column 1. From the start, you can move up to (4,2), which is open. From there, continue exploring valid paths. Re-examine the grid using the correct positions and trace possible moves step by step. Focus on building a path using only open cells (O), avoiding obstacles (X).

Second Attempt:
Step 1: Correctly identify the start and destination coordinates
The start (*) is at row 5, column 2 (0-indexed), and the destination (#) is at row 2, column 1.
Step 2: Analyze possible movements from the start cell
From the start at (5,2), move up to (4,2), which is an open cell (O).
Step 3: Continue exploring valid paths
From (4,2), we can move up to (3,2), which is also an open cell (O).
Step 4: Further explore valid paths from (3,2)
Moving up from (3,2) takes us to (2,2), another open cell (O).
Step 5: Determine the feasibility of reaching the destination
The destination (#) at (2,1) is reachable by moving left from (2,2).
Step 6: Conclusion on the path from start to destination
It is possible to reach the destination (#) from the start (*) by moving up from (5,2) to (4,2), then to (3,2), and finally to (2,2), followed by a valid left move to (2,1).
The final answer is: up, up, up, left

Shortest Path Example (Step 616)

Question:
Find the shortest path from the start (*) to the destination (#) in this grid. You can only move through open cells (O) and cannot pass through obstacles (X). Valid moves are: right, down, left, up.

Grid:
O X X O O O
X O X O O #
X * X X O X
X O O O O O
O O X O X X
O O O O X O
If a path exists, return it as a sequence of moves (e.g., "right, down, right"). If no path exists, return "infeasible". Let's think step by step and output the final answer within boxed{}.

First Attempt:
Step 1: Identify the Start Location
The start (*) is located at row 2, column 1 on the grid.
Step 2: Identify the Destination
The destination (#) is located at row 1, column 5 on the grid.
Step 3: Explore the Grid from the Start
From row 2, column 1, we begin by examining the cell directly to the right. We find that row 2, column 2 is an X and cannot be visited. We continue by examining the cell directly below row 2, column 1. We find that row 3, column 1 is an O and can be visited. From row 3, column 1, we continue by examining the cell directly to the right. We find that row 3, column 2 is an O and can be visited. From row 3, column 2, we continue by examining the cell directly to the right. We find that row 3, column 3 is an O and can be visited. From row 3, column 3, we continue by examining the cell directly to the right. We find that row 3, column 4 is an O and can be visited. From row 3, column 4, we continue by examining the cell directly above. We find that row 2, column 4 is an O and can be visited. From row 2, column 4, we continue by examining the cell directly above. We find that row 1, column 4 is an O and can be visited. From row 1, column 4, we continue by examining the cell directly to the right. We find that row 1, column 5 is # and is the destination.
Step 4: Conclusion
We conclude that the path from the start (*) at row 2, column 1 involves continuing by examining the cell directly below to row 3, column 1, followed by row 3, column 2, row 3, column 3, row 3, column 4, row 2, column 4, row 1, column 4, and finally row 1, column 5.

Below, we also show example generations from `Knights and Knaves` and `DeepMath`.

Knights and Knaves Example (Step 89)

Question:
A very special island is inhabited only by sages and fools. Sages always tell the truth, and fools always lie. You meet 5 inhabitants: Lucas, Avery, Sebastian, Emily, and Elizabeth. "it is not the case that (Avery is a sage and Emily is a sage)," Lucas mentioned. "Lucas is a sage," Avery declared. Sebastian said, "if (Sebastian is a sage or Elizabeth is a fool) then (Avery is a sage and Sebastian is a sage)." "(if Lucas is a fool then Emily is a sage) if and only if Sebastian is a sage," Emily declared. "(Sebastian is a fool or Elizabeth is a sage or Emily is a fool) and (Sebastian is a fool if and only if Lucas is a fool) and Avery is a sage," Elizabeth declared. So who is a sage and who is a fool? (Format your answer like: "Lucas is a sage/fool, Avery is a sage/fool, Sebastian is a sage/fool, Emily is a sage/fool, and Elizabeth is a sage/fool") Let's think step by step and output the final answer within boxed{}.

First Attempt:
Step 1: Analyze Lucas's statement Lucas said, "it is not the case that (Avery is a sage and Emily is a sage)." This statement implies that either Avery is not a sage or Emily is not a sage (or both).
Step 2: Analyze Avery's statement Avery declared, "Lucas is a sage." Since sages always tell the truth, Avery must be a sage because he is correctly identifying Lucas as a sage.
Step 3: Analyze Sebastian's statement Sebastian said, "if (Sebastian is a sage or Elizabeth is a fool) then (Avery is a sage and Sebastian is a sage)." Given that Avery is a sage from Avery's statement, for Sebastian's conditional statement to hold true (and be truthfully revealed by

a sage who is a side of the or condition), Sebastian must also be a sage (since being a fool would require him to lie).
Step 4: Analyze Emily's statement Emily declared, "(if Lucas is a fool then Emily is a sage) if and only if Sebastian is a sage." Since Sebastian is a sage from step 3, the if and only if part is relevant. Given that Sebastian is a sage, for the statement to be true (as it is made by a sage), the premise about Lucas being a fool implying Emily is a sage must be false. Thus, Lucas is not a fool.
Step 5: Analyze Elizabeth's statement Elizabeth declared, "(Sebastian is a fool or Elizabeth is a sage or Emily is a fool) and (Sebastian is a fool if and only if Lucas is a fool) and Avery is a sage." Given that Avery is indeed a sage from step 2, the final part of the statement is confirmed to be true. The first part, with all the conditions, would require at least one of the conditions involving Sebastian or Elizabeth or Emily being a fool, but since we've established Lucas is not a fool, and Sebastian is a sage from step 3, the statement holds true for any of these being a fool. However, given all the previous deductions, the direct implication from this statement doesn't necessarily reveal new truth but solidifies existing truths.
Step 6: Summarize the deductions - Lucas is a sage because he's not presented as a liar and his statement about Avery and Emily is factually considered in relation to other statements. - Avery is a sage, confirmed by the truthfulness of his statement. - Sebastian is a sage, deduced from the conditional statement and its implications on his own truthfulness. - Emily is a sage because her statement about the relation of Sebastian being a sage and the implication on Lucas needing to be a fool for her to be a sage is not relevant given Sebastian's truth status. - Elizabeth is a fool because her statement, given as a sage, requires conditions that could be fulfilled by lies within her own statement, and given the deductions made are based on the truthfulness of others. The final answer is: boxed{Lucas is a sage, Avery is a sage, Sebastian is a sage, Emily is a sage, and Elizabeth is a fool}

Expert Critique:
Your previous attempt was incorrect. The major error is assuming Avery is a sage without proof—this is circular. You cannot assume Avery is a sage just because he says Lucas is one. Similarly, assuming Sebastian is a sage based on the truth of the consequent is invalid—remember that the truth of a conditional depends on both truth value and structure. The biconditional in Emily's statement must be evaluated more carefully: if Sebastian is a fool, the biconditional requires the implication (if Lucas is fool $\rightarrow$ Emily is sage) to be false. Try testing assumptions with truth tables or counterfactuals. Identify which statements force contradictions if certain roles are assumed. Focus on Elizabeth's statement—it's long but has a "and" structure, so for a fool, *at least one* part must be false. This can help eliminate possibilities.

Second Attempt:
Step 1: Initial Analysis We begin by analyzing the statements given by each inhabitant of the island, remembering that sages always tell the truth, and fools always lie. Our goal is to determine who is a sage and who is a fool.
Step 2: Analyzing Lucas's Statement Lucas says, "it is not the case that (Avery is a sage and Emily is a sage)." This statement implies that either Avery is not a sage or Emily is not a sage (or both).
Step 3: Analyzing Avery's Statement Avery declares, "Lucas is a sage." To determine if Avery is a sage or a fool, we must evaluate his statement in the light of Lucas's character. We cannot immediately assume Avery is a sage because that would be based on his own claim, which may or may not be true.
Step 4: Analyzing Sebastian's Statement Sebastian says, "if (Sebastian is a sage or Elizabeth is a fool) then (Avery is a sage and Sebastian is a sage)." This conditional statement necessitates a closer look. The antecedent (Sebastian is a sage or Elizabeth is a fool) cannot be evaluated immediately because we need to determine Sebastian's character.
Step 5: Analyzing Emily's Statement Emily says, "(if Lucas is a fool then Emily is a sage) if and only if Sebastian is a sage." This statement involves a biconditional and must be evaluated carefully. If Sebastian were a fool, for Emily to be speaking the truth, the implication (if Lucas is fool $\rightarrow$ Emily is sage) must be false, but since we cannot determine Sebastian's character yet, we must proceed with caution.

Step 6: Analyzing Elizabeth's Statement Elizabeth says, "(Sebastian is a fool or Elizabeth is a sage or Emily is a fool) and (Sebastian is a fool if and only if Lucas is a fool) and Avery is a sage." Given the "and" structure, for Elizabeth to be a fool, at least one part of her statement must be false. However, we need to see how Avery's truthfulness affects this analysis.
Step 7: Evaluating Truth Conditions We must evaluate each statement under the condition that the speaker might be either a sage or a fool. Lucas's statement does not depend on the truth of others in a direct linear fashion but rather sets the scene for the entire puzzle. Avery's statement can only be considered under the separation of Lucas being a sage or not due to the nature of the puzzle's setup.
Step 8: Truth Table Approach Given the complexity of the statements, a truth table approach could help. However, without specific conditions or values assigned to Lucas, Avery, Sebastian, Emily, and Elizabeth, and without the ability to directly evaluate their truthfulness based on their statements alone without circular reasoning, we must focus on the logical structure of the statements they made.
Step 9: Focusing on Logical Implications We should identify which statements, if assumed, would force contradictions. Key implications include Elizabeth's statement having two distinct components, one of which involves the relationship between Sebastian and Lucas and another related to their characters and the truth of Avery's sagehood.
Step 10: Identifying Constraints Given the structure of the statements, if we can establish that there is at least one condition (from Elizabeth's statement) that Elizabeth, as a potential fool, must have false, this could give us a concrete path towards determining who is a sage and who is a fool, considering the truthfulness of the other inhabitants.
Step 11: Contradiction Test Let's directly test for contradictions by making assumptions about who is a sage or a fool and determining if each inhabitant's statement would hold true or not under those assumptions.
Step 12: Establishing Truth and Falsehood Through process of elimination and evaluating the logical implications of the statements, especially considering the "and" structure of Elizabeth's statement, we can establish who must be a sage or a fool.
Step 13: Finalizing Roles Based on the logical analysis and the conditions set forth by each inhabitant, we can assign roles of sage or fool to each of the individuals.
Step 14: Conclusion After analyzing the statements and their logical implications, we can now determine who is a sage and who is a fool among the inhabitants of the island. The final answer is: boxed{Lucas is a sage, Avery is a sage, Sebastian is a fool, Emily is a fool, and Elizabeth is a fool}

Deepmath Example (Step 1290)

Question:
Let $x_1, x_2, \ldots, x_{1970}$ be positive integers such that $x_1 + x_2 + \cdots + x_{1970} = 2007$. Find the largest possible value of $x_1^3 + x_2^3 + \cdots + x_{1970}^3$. Let's think step by step and output the final answer within boxed{}.

First Attempt:
Step 1: Understand the problem We are given a sequence of 1970 positive integers, $x_1, x_2, \ldots, x_{1970}$, such that their sum is 2007. We need to find the largest possible value of the sum of the cubes of these integers, i.e., $x_1^3 + x_2^3 + \cdots + x_{1970}^3$.
Step 2: Use the concept of weighted inequalities A common approach to problems of this type is to use inequalities that relate the sum of powers to the sum of the numbers, such as Cauchy-Schwarz or inequalities involving the sum of cubes. One powerful tool is Cauchy-Schwarz, but here we might consider inequalities that compare sums of integers to sums of their cubes, such as $\sum x_i^3 \geq \frac{(\sum x_i)^3}{1970}$, although we must carefully consider how to apply such inequalities and whether they give the maximum.
Step 3: Recall the power mean inequality The power mean inequality can be useful here, particularly the inequality that compares the arithmetic mean and the cubic mean. However, our main concern is maximizing $\sum x_i^3$, subject to $\sum x_i = 2007$. A more direct approach involves focusing on how to distribute the sum of 2007 across the 1970 integers to maximize the total cubic sum.

Step 4: Consider the equality case for weighted sums For inequalities that relate sums of powers, such as $\sum x_i^3 \geq \frac{(\sum x_i)^3}{n}$, equality occurs when all x_i are equal. However, since we are looking to maximize this sum under a constraint and the numbers must be positive integers, we cannot set all $x_i = 1$, as that would sum to 1970, not 2007. Instead, we aim to allocate as much of the sum as possible to the largest x_i, because x^3 grows faster than x, so concentrating the sum in the largest possible value will generally increase the cube sum more than spreading it out.
[Skip parts of irrelevant generation]
Step 29: Evaluate the cube of $x_i = 1$ If all $x_i = 1$, then $\sum x_i^3 = 1970 \cdot 1^3 = 1970$.
Step 30: Use the fact that the sum of cubes is maximized by making one x_i as large as possible We seek to make one x_i as large as possible while ensuring all are positive integers and the sum is 2007. If we set all but one $x_i = 1$, that leaves 2007 - 1969 = 38 to be assigned to the remaining x_i. So, we could set one $x_i = 38$ and the rest = 1.
Step 31: Calculate the sum of cubes for $x_i = 1$ and one $x_i = 38$ If one $x_i = 38$ and the rest = 1, then the sum of cubes is $1^3 + 1^3 + \cdots + 1^3 + 38^3$, which is $1969 \cdot 1^3 + 38^3$.
Step 32: Perform the calculation of 38^3 $38^3 = 38 \times 38 \times 38 = 54,872$.
Step 33: Add 1969 to the result of 38^3 $54,872 + 1969 = 57,841$.
Step 34: Evaluate the total sum of cubes The total sum of cubes is $1969 + 57,841 = 59,810$.
Step 35: Compare this to the sum when all $x_i = 1$ When all $x_i = 1$, the sum of cubes is 1970. Our calculated sum is 59,810, which is larger.
Step 36: Confirm the maximum Based on the calculation, the maximum possible value of $\sum x_i^3$ under the given constraints is 59,810. The final answer is: $\boxed{59110}$

Expert Critique:
Your approach to maximize the sum of cubes by concentrating value in one variable is correct and well-reasoned. However, you made an arithmetic error when computing the total sum of cubes: $54,872 + 1,969 = 56,841$, not 59,110. Double-check your additions, especially when combining large numbers. Your allocation of one 38 and 1969 ones is valid and optimal under the constraints. The error is purely computational. Recheck the sum: $38^3 = 54,872$, plus 1969 ones contributes 1,969. Add them carefully. Your previous attempt was incorrect.

Second Attempt:
Step 1: Understand the problem We are given 1970 positive integers, denoted as $x_1, x_2, \ldots, x_{1970}$, with the constraint that their sum equals 2007. Our goal is to find the largest possible value of the sum of their cubes, i.e., $x_1^3 + x_2^3 + \cdots + x_{1970}^3$.
Step 2: Analyze the constraint and objective The constraint is $x_1 + x_2 + \cdots + x_{1970} = 2007$, and we want to maximize $x_1^3 + x_2^3 + \cdots + x_{1970}^3$. Since the cubic function $f(x) = x^3$ is strictly increasing for positive x, the maximum of the sum of cubes will occur when the values are concentrated in as few variables as possible, with the largest possible value assigned to one variable.
Step 3: Reason about the distribution To maximize the sum of cubes, we should make one integer as large as possible and the rest as small as possible, while still satisfying the sum constraint. This is because the cube function grows very rapidly, so even a small increase in one variable can lead to a much larger increase in its cube.
Step 4: Calculate the maximum possible value We want to assign the largest possible value to one variable, say x_1. The maximum possible value for x_1 occurs when we set all other variables to the smallest possible value, which is 1. The sum constraint tells us that $x_1 + 1969 \cdot 1 = 2007$, so we solve for x_1: $x_1 = 2007 - 1969 = 38$.
Step 5: Calculate the sum of cubes Now we calculate the sum of cubes under this allocation: $x_1^3 + x_2^3 + \cdots + x_{1970}^3 = 38^3 + 1969 \cdot 1^3$. First, compute $38^3 = 38 \times 38 \times 38 = 54,872$. Then, multiply $1969 \times 1^3 = 1969$. Finally, add these two values together.
Step 6: Perform the addition carefully Now we add $54,872 + 1969$ carefully. Starting with $54,000 + 1969 = 55,969$, then add the remaining 892 to get $56,861$. No, we must do it precisely: $54,872 + 1,969 = 56,841$.
Step 7: Confirm the optimal allocation We have found that the optimal allocation is to set one variable to 38 and the rest to 1. This is valid under the constraints, and the sum of cubes under this allocation is $56,841$.

Step 8: Conclude the maximum sum of cubes Therefore, the largest possible value of the sum of cubes $x_1^3 + x_2^3 + \cdots + x_{1970}^3$ is $56,841$. The final answer is: $\boxed{56841}$.

D.3 Details of Benchmarks

In this subsection, we describe our evaluation setup for each domain.

Reasoning. Reasoning Gym (Stojanovski et al., 2025) is a library of reasoning puzzles spanning categories such as algebra, arithmetic, games, geometry, graphs, induction, and logic. We evaluate on three tasks from Reasoning Gym: `Knights and Knaves`, `Binary Matrix`, and `Shortest Path`. `Knights and Knaves` is a logic puzzle where some characters are truthful "knights" and the rest are untruthful "knaves", and the goal is to deduce from the characters' statements whether they are knights or knaves. `Binary Matrix` is an algorithmic problem, where the input is a square matrix of 1's and 0's and the task is to find the Manhattan distance to the nearest 0 for each cell. `Shortest Path` is a graph problem, where the input is a matrix containing `*`, `#`, `O`, and `X`, representing the start cell, the destination cell, an open cell, and a blocked cell, respectively. The objective is to find the shortest path from the start to the destination, using only open cells. For all environments, we procedurally generate 20K problems in similar style to the Reasoning Gym code [4]. We use 19.8K examples for training and the remaining 200 examples for testing.

Math. We evaluate on two standard math benchmarks, MATH500 and AIME24. MATH500 (Hendrycks et al., 2021) is a competition math dataset of 500 problems spanning seven categories: algebra, counting and probability, geometry, intermediate algebra, number theory, prealgebra and precalculus. The American Invitational Mathematics Examination (AIME) is a prestigious high school math competition. There are two version of the exam with 15 questions each, for a total of 30 questions in the 2024 test set. We consider training on two training sets: DAPO-17K (Yu et al., 2025), which contains 17K problems sourced from math competitions, and DeepMath-103K (He et al., 2025), which contains 103K problems that are of generally higher difficulty than DAPO-17K.

Creative writing. LitBench (Fein et al., 2025) is a dataset containing 43K pairwise examples of LLM-generated stories and human preference labels. We train on the LitBench train set and evaluate on the test set of 2K examples. We also use the same checkpoint and evaluate on WritingBench (Wu et al., 2025), which is a benchmark containing 1K real-world writing tasks, spanning academics and engineering, finance and business, politics and law, literature and art, education, and advertising and marketing. For evaluation, we use the prompt-specific rubric from WritingBench and GPT-4.1-mini as the judge.

[4] https://github.com/open-thought/reasoning-gym

D.4 HYPERPARAMETERS

Hyperparameters specific to Reasoning Gym environment configurations:

- Knights and Knaves: `n_people=5, depth_constraint=3, width_constraint=3`
- Binary Matrix: `min_n=3, max_n=5`
- Shortest Path: `min_rows=5, min_cols=5, max_rows=6, max_cols=6, p_blocked=0.4`

Table 2: Hyperparameters for all multi-turn RL experiments.

Hyperparameter	**Value**
Group size	8
Groups per batch	32
Max tokens	8096
LoRA rank	32
Learning rate	2e-5
KL penalty	0.0
RL coefficient (Self Distillation)	0.1
SFT coefficient (Feedback Modeling)	0.2

For math training, we use a linear schedule for the RL coefficient from 0 to 0.1.

Feedback Descent. There is no publicly available codebase for Feedback Descent, so we reimplement the algorithm based on details from the Feedback Descent (Lee et al., 2025) paper and the GEPA (Agrawal et al., 2025) paper. For the candidate comparison prompt, we use "System Prompt Template for Prompt Optimization" from the Feedback Descent paper. Across all tasks, we perform 5 rounds of prompt improvement on the training set, collect all candidate prompts encountered during optimization, evaluate them on a held-out validation set, and select the best-performing prompt for final evaluation on the test set. We used a temperature of 0.6 for Reasoning Gym and competition math and a temperature of 0.7 for creative writing. For Reasoning Gym tasks, we use 200 training and 200 validation examples. For the competition math setups, we use 64 training and 16 validation examples. For creative writing, we use 20 training and 16 validation examples. We perform evaluations on the full test set. While these training sets are smaller than the fully available training data, we aimed to follow the precedent of prior prompt optimization work (e.g., GEPA uses on the order of 100-150 examples for training) while using a reasonable amount of compute for our budget. Prompt optimization is generally highly sample-efficient, and we also observed empirically that increasing the number of training and validation examples did not yield meaningful performance improvements.

D.5 VARIANCE OF IMPORTANCE WEIGHTING

Let $q(\cdot)$ denote the second-turn (teacher) distribution over outputs, induced by one round of feedback: $y_1 \sim q(\cdot) := \pi(\cdot \mid x_1)$ with $x_1 = f(x_0, y_0, c_0)$. Recall that an unbiased estimator of the single-turn policy gradient can be obtained by importance weighting with $w(y) = \pi(y \mid x_0)/q(y)$, but its variance is controlled by the second moment $\mathbb{E}_q[w^2]$. In particular, $\mathbb{E}_q[w^2] = 1 + \chi^2(\pi(\cdot \mid x_0)\|q) = \exp(D_2(\pi(\cdot \mid x_0)\|q))$ [5], thus unlike the commonly used KL divergence, the variance is measured with directly density ratio instead of the logarithm of the density ratios.

To measure the variance empirically, we consider the `AIME24` task, and we measure the average trajectory-level log ratio and average token-level log ratio between the first-turn and second-turn policies, and we summarize the results in Table 3. Note that the trajectory-level log ratio is almost vacuously large (recall that the variance if measure on the direct policy ratio, thus the exponential of the log ratio), indicating that importance weighting at the trajectory level is infeasible. On the other hand, the token-level log ratio also exhibits high variance, indicating long tail behavior. This explains the difficulty of applying importance weighting in practice.

Table 3: Average log importance weight ratios between first-turn and second-turn policies on `AIME24`.

	Value
Average (Trajectory)	142.1637
Average (Token)	0.1288
Standard Deviation (Token)	0.5506

[5] For probability measures p and q with $p \ll q$, the χ^2-divergence and order-2 Rényi divergence are $\chi^2(p\|q) = \int \frac{(p(y)-q(y))^2}{q(y)}\, dy$, and $D_2(p\|q) = \log \int \frac{p(y)^2}{q(y)}\, dy$.

Table 4: Comparison of baselines across **reasoning puzzles**, **competition math**, and **creative writing** tasks. We report second-turn accuracy after 2-turn training (i.e., $J_{\mathsf{SingleTurn}}(\pi)$) of the last checkpoint. For the reasoning tasks and `LitBench`, we report the mean@1 accuracy, judged by either verifiable reward or LLM-as-a-judge. For the math tasks, we report the mean@32 accuracy from the last checkpoint from the training. The parentheses denote the training dataset. For `WritingBench`, we follow the official protocol with GPT-4.1-mini as the judge. The accuracy in reasoning and math is normalized between 0 and 1, and the score in creative writing is normalized between 1 and 10. Note that Self Distillation and Feedback Modeling consistently outperform all baselines across tasks.

	Base Model	GRPO 2-turn	RLTF-SD	RLTF-FM
Reasoning				
`Knights and Knaves`	0.119	0.636	0.896	0.910
`Binary Matrix`	0.001	0.978	0.989	0.993
`Shortest Path`	0.034	0.836	0.892	0.976
Math				
`Math500 (DAPO)`	0.583	0.692	0.725	0.771
`AIME24 (DAPO)`	0.042	0.142	0.185	0.250
`Math500 (Deepmath)`	0.583	0.741	0.765	0.793
`AIME24 (Deepmath)`	0.042	0.133	0.203	0.208
Creative writing				
`LitBench`	5.84	8.02	9.20	8.60

D.6 Additional Results

Here, we provide evaluation curves for our general results in reasoning, math and creative writing, corresponding to Tables 1 and 4. Figure 5 shows the single-turn accuracy and Figure 6 shows the multi-turn accuracy.

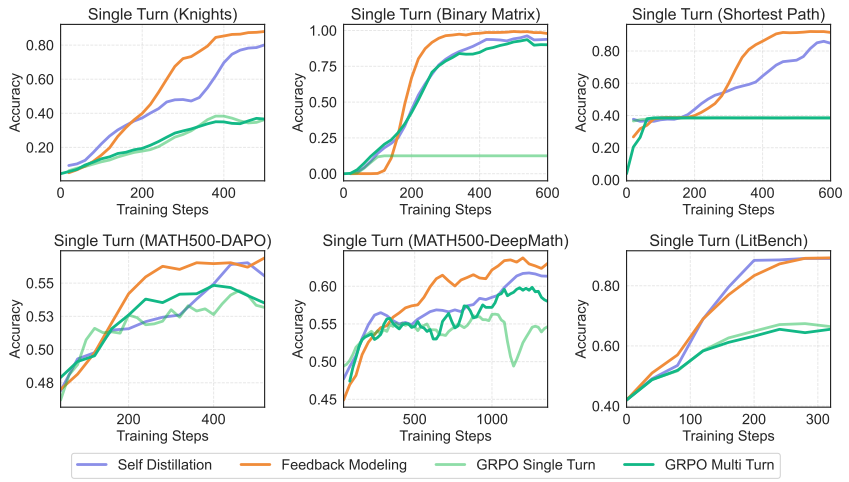

Figure 5: Evaluation curves for single-turn accuracy across **reasoning puzzles**, **competition math**, and **creative writing** tasks, at every 40 training steps. For reasoning tasks, we report the mean@1 accuracy judged by either verifiable reward or LLM-as-a-judge. For math tasks, we report the mean@1 accuracy. For `Litbench`, we report the mean@1 accuracy judged by LLM-as-a-judge. The accuracy in reasoning and math is normalized between 0 and 1, and the score in creative writing is normalized between 1 and 10.

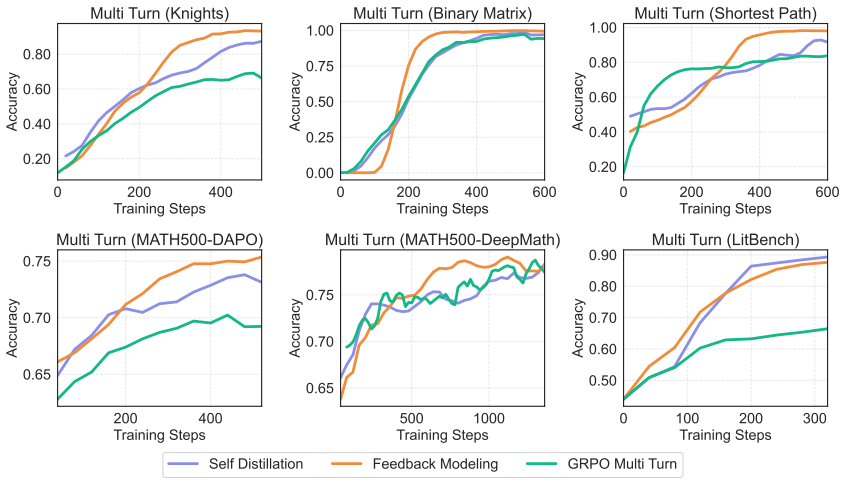

Figure 6: Evaluation curves for multi-turn accuracy across **reasoning puzzles**, **competition math**, and **creative writing** tasks, at every 40 training steps. For reasoning tasks, we report the mean@1 accuracy judged by either verifiable reward or LLM-as-a-judge. For math tasks, we report the mean@1 accuracy. For `Litbench`, we report the mean@1 accuracy judged by LLM-as-a-judge. The accuracy in reasoning and math is normalized between 0 and 1, and the score in creative writing is normalized between 1 and 10.

D.7 Resources

All of our training experiments are performed using Tinker (Lab, 2025).