LANGUAGE MODELS CAN LEARN FROM VERBAL FEEDBACK WITHOUT SCALAR REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

LLMs are often trained with RL from human or AI feedback, yet such methods typically *compress nuanced feedback into scalar rewards*, discarding much of their richness and inducing scale imbalance. We propose treating verbal feedback as a conditioning signal. Inspired by language priors in text-to-image generation, which enable novel outputs from unseen prompts, we introduce the **feedback-conditional policy (FCP)**. FCP learns directly from response-feedback pairs, approximating the feedback-conditional posterior through maximum likelihood training on *offline* data. We further develop an *online bootstrapping* stage where the policy generates under positive conditions and receives fresh feedback to refine itself. This reframes feedback-driven learning as conditional generation rather than reward optimization, offering a more expressive way for LLMs to directly learn from verbal feedback.

1 INTRODUCTION

The *reward hypothesis* in reinforcement learning (RL) was proposed over two decades ago, when feedback from the environment had to be reduced to **scalar rewards** for RL algorithms to operate (Sutton, 2004). This view shaped much of the field's progress and remains the prevailing standard in applying RL to *alignment* and *reasoning* for large language models (LLMs) (Ziegler et al., 2019; Bai et al., 2022; Rafailov et al., 2023; Guo et al., 2025).

Yet in practice, feedback encountered in RL for LLMs, especially from generative reward models and LLM critics (Lee et al., 2023; Zhang et al., 2024; Mahan et al., 2024; Ankner et al., 2024), is originally **verbalized**, such as "`Vivid imagery, but the character arc lacks depth`". Rather than utilizing this rich semantic critique, standard pipelines typically compress it into a scalar reward for RL optimization. Reducing verbal feedback into scalar rewards introduces several limitations:

- **I. Information loss.** Scalar rewards capture far less information than verbal feedback and are often uninterpretable. For example, the critiques "`The response is redundant but correct`" and "`The response is compact but has many typos`" may both collapse to a reward of 0.8, despite describing very different response patterns. Furthermore, the verbalized thoughts produced by generative reward models are typically discarded, with only the final scalar retained for RL training.
- **II. Ambiguity.** Verbal feedback, especially in complex or multi-dimensional tasks, is often *mixed* (containing both pros and cons) or *uncertain*, such as "`I cannot fully verify whether step 3 holds in general.`". Such feedback is common and carries rich cues across complementary criteria. Mapping these multi-faceted critiques into a single scalar is inherently ambiguous or arbitrary.
- **III. Imbalanced reward scales across tasks.** In multi-task training (e.g., math, science, writing), it is difficult to maintain a consistent reward scale. Positive feedback on a simple math problem is far easier to obtain than on a challenging scientific task, which induces imbalanced rewards across domains and biases the learning process.

Scalarization has long been seen as unavoidable, bridging verbal feedback and the numerical signals required by RL. With the rise of large-scale language pretraining, however, this view is being re-examined (Yao, 2025). LLMs embody strong commonsense and linguistic **priors**, suggesting a new paradigm: *treat verbal feedback as a first-class training signal*, rather than forcing it into a scalar form.

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

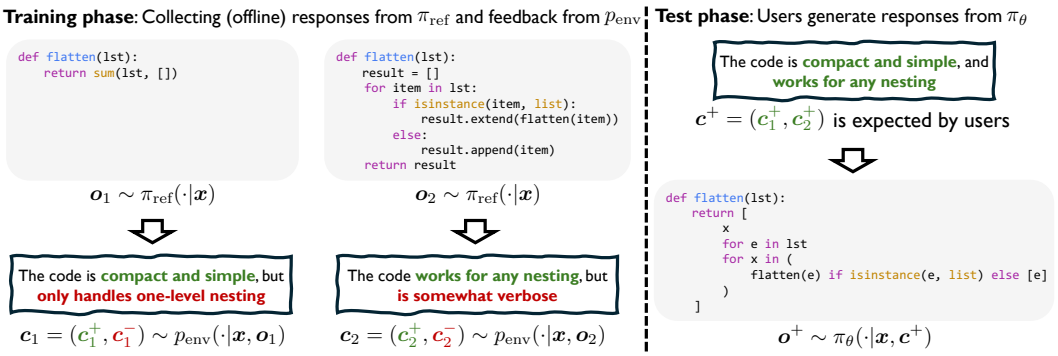


Figure 1: **Learning from mixed verbal feedback.** The instruction x is "*Write a Python function flatten(lst) that returns a flat list of integers*". The reference policy π_{ref} may assign low probability to the ideal response o^+, making purely positive response-feedback pairs (o^+, c^+) rare in the training data collected from π_{ref} and p_{env}. This resembles the setting of text-to-image generation, where the *language prior* enables models to combine seen captions (analogous to mixed feedback c_1 and c_2) and generate rare images (analogous to purely positive response o^+) such as "*a banana surfing on the ocean*" (Figure 2). Motivated by this, our model π_θ is trained as a feedback-conditional policy (FCP), and when conditioning on desired positive c^+, we have $\pi_\theta(o|x, c^+) \propto \pi_{\text{ref}}(o|x) \cdot p_{\text{env}}(c^+|x, o)$.

After all, LLMs already show the ability to **implicitly understand verbal feedback**. In iterative drafting and revision, models readily adapt by incorporating editorial critiques into their context and refining their prose accordingly (Chen et al., 2024). This indicates that LLMs can process verbal feedback, but only implicitly, *through an in-context mechanism that does not convert understanding into explicit scalar rewards*. The key question, then, is how to distill such feedback into training so that it directly improves model performance, rather than relying on multi-turn trial and error at test time.

To this end, we propose to learn a **feedback-conditional policy (FCP)** $\pi_\theta(o|x, c) \propto \pi_{\text{ref}}(o|x) \cdot p_{\text{env}}(c|x, o)$, where $\pi_{\text{ref}}(o|x)$ is a reference policy that generates a response o given an instruction x, and $p_{\text{env}}(c|x, o)$ is the distribution of environment feedback c. Intuitively, the FCP reweights the reference policy by how likely each response o would elicit the observed feedback c. Conditioning on positive feedback c^+ gives $\pi_\theta(o|x, c^+) \propto \pi_{\text{ref}}(o|x) \cdot p_{\text{env}}(c^+|x, o)$, which increases the probability of generating responses that are more likely to receive favorable feedback. In this way, the FCP learns a *posterior* distribution that integrates prior knowledge from π_{ref} with verbal feedback, allowing it to handle diverse forms of feedback, as illustrated in Figure 1.

After training an *offline* FCP $\pi_\theta(o|x, c) \propto \pi_{\text{ref}}(o|x) \cdot p_{\text{env}}(c|x, o)$ that conditions on arbitrary feedback c, we further improve it via *online bootstrapping*. Concretely, we conduct online training by sampling rollouts from the behavior policy $\pi_\theta(o|x, c^+)$ (goal-conditioned on positive feedback), and re-annotating them with fresh feedback from p_{env}, thereby iteratively strengthening the policy.

As a proof-of-concept across both verifiable reasoning tasks and open-ended creative writing with multi-attribute critiques, our experiments show that FCP matches or surpasses strong scalar-based baselines such as offline RFT (Dong et al., 2023) and online GRPO (Shao et al., 2024), **without relying on verifiers, scalar conversion, or data filtering**. This demonstrates a simple and scalable framework that preserves the richness of verbal feedback while avoiding the scarcity of rule-based verifiers.

Throughout, we focus on verbal feedback produced by LLM critics, the most scalable source of verbal feedback in current RL pipelines for LLMs, which also allows controlled comparisons between verbal and scalar signals derived from the *same* critic. We further probe robustness to less curated feedback by varying critique style and quality (Section 4.1), and leave training on in-the-wild human feedback to future work.

2 Learning Directly from Verbal Feedback

Traditional RL trains a policy by up-weighting responses that receive "good" feedback and down-weighting those that receive "bad" feedback. From a probabilistic view, RL can be seen as learning a *posterior* over responses that are expected to receive good feedback (i.e., high rewards) (Peters & Schaal, 2007; Peng et al., 2019; Rafailov et al., 2023). Distinguishing what counts as good or bad

Algorithm 1 Offline training: Initializing feedback-conditional policy (Section 2.1)

Inputs: Reference policy $\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})$, feedback environment $p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})$, feedback-conditional policy $\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})$, instruction corpus $\mathcal{X}$, batch size B, optimizer $\mathcal{O}$
Outputs: The offline-trained parameters θ_{off}

1: **Initialize** $\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}) = \pi_{\text{ref}}\left(\boldsymbol{o}\middle|\left[\texttt{<EF>}\boldsymbol{c}\texttt{</EF>},\boldsymbol{x}\right]\right)$, where <EF> and </EF> are special tokens used to wrap the expected feedback $\boldsymbol{c}$, which is concatenated before the instruction $\boldsymbol{x}$
2: **Collect** offline dataset $\mathcal{D}_{\text{off}} = \{(\boldsymbol{x},\boldsymbol{o},\boldsymbol{c})\}$ with $\boldsymbol{x}\sim\mathcal{X}$, $\boldsymbol{o}\sim\pi_{\text{ref}}(\cdot|\boldsymbol{x})$ then $\boldsymbol{c}\sim p_{\text{env}}(\cdot|\boldsymbol{x},\boldsymbol{o})$
3: **Objective:** $\max_\theta \mathbb{E}_{(\boldsymbol{x},\boldsymbol{o},\boldsymbol{c})\sim\mathcal{D}_{\text{off}}}[\log \pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})]$ // Taking expectation over $x\sim\mathcal{X}$ in Eq. (1)
4: **while** not converged **do**
5: Sample $\{(\boldsymbol{x}_i,\boldsymbol{o}_i,\boldsymbol{c}_i)\}_{i=1}^B \sim \mathcal{D}_{\text{off}}$; $\theta \leftarrow \mathcal{O}.\text{step}\big(\theta, \nabla_\theta \frac{1}{B}\sum_{i=1}^B \log \pi_\theta(\boldsymbol{o}_i|\boldsymbol{x}_i,\boldsymbol{c}_i)\big)$
6: **return** $\theta_{\text{off}} \leftarrow \theta$

typically requires carefully designed reward functions or detailed rubrics to produce scalar signals, leading to the limitations discussed in Section 1.

Our approach is inspired by language priors in text-to-image generation, where models compose *unseen* prompts from mixed captions (Figure 2). Similarly, language priors could enable LLMs to absorb diverse verbal feedback and yield high-quality responses beyond scalar reinforcement (Figure 1). Since LLMs already show implicit feedback understanding, we train directly on it: *offline* to initialize a **feedback-conditional policy (FCP)** (Section 2.1), then *online* to bootstrap performance (Section 2.2).

2.1 Offline Training: Initializing FCP

We begin with a reference policy model π_{ref} that takes an input instruction $\boldsymbol{x}$ and generates a response $\boldsymbol{o}\sim\pi_{\text{ref}}(\cdot|\boldsymbol{x})$. The response $\boldsymbol{o}$ then undergoes a *single-turn* interaction with the environment, which provides verbal feedback $\boldsymbol{c}\sim p_{\text{env}}(\cdot|\boldsymbol{x},\boldsymbol{o})$. The reference policy π_{ref} may represent a base model, an instruction-tuned model, or a reasoning model, and the response $\boldsymbol{o}$ can include both thinking processes and the final answer. The environment p_{env} can be any black-box feedback source; in this paper it is instantiated by generative reward models or LLM critics, though the formulation applies equally to human annotators. In the **offline** setting, where responses are collected from π_{ref}, we define the joint distribution of response-feedback pairs as $P_{\text{off}}(\boldsymbol{o},\boldsymbol{c}|\boldsymbol{x}) \triangleq \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})\cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})$, from which we derive the *feedback-conditional posterior* distribution:

$$P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}) = \frac{P_{\text{off}}(\boldsymbol{o},\boldsymbol{c}|\boldsymbol{x})}{P_{\text{off}}(\boldsymbol{c}|\boldsymbol{x})} = \frac{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})\cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})}{\sum_{\boldsymbol{o}}\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})\cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})}.$$

Informally, let $\boldsymbol{c}^+$ denote purely positive feedback and $\boldsymbol{c}^-$ purely negative one. Mixed feedback can be approximated as $\boldsymbol{c}=(\boldsymbol{c}^+,\boldsymbol{c}^-)$, while neutral or uncertain feedback may be neither. If we condition on positive feedback $\boldsymbol{c}^+$, for instance, "`The generated code is functionally correct and efficient`" for a coding instruction $\boldsymbol{x}$, then $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}^+) \propto \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})\cdot p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x},\boldsymbol{o})$, which favors responses $\boldsymbol{o}$ that are more likely to elicit positive feedback.[1]

While $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}^+)$ is the policy we are seeking, it cannot be sampled directly: $p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x},\boldsymbol{o})$ is only defined after a full response is generated, and for black-box critics its likelihood cannot be evaluated at all. This posterior is also the optimum of KL-regularized RL with reward $\log p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x},\boldsymbol{o})$ (Peters & Schaal, 2007; Rafailov et al., 2023) (Appendix B.1), but solving that objective requires exactly this intractable likelihood, so *we do not optimize it*. Instead, we fit the posterior directly by minimizing the *forward* KL divergence, which, as shown next, only requires samples from π_{ref} and p_{env}. Moreover, since feedback cannot be restricted to be positive without rubrics or filtering, we fit $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})$ for *any* feedback $\boldsymbol{c}$. This does not change the target at $\boldsymbol{c}^+$: the optimum still recovers $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}^+)$ as long as $P_{\text{off}}(\boldsymbol{c}^+|\boldsymbol{x})>0$ (see below).

Specifically, we propose to learn a **feedback-conditional policy (FCP)** $\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})$ by minimizing the expected forward KL divergence between $\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})$ and $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})$:

$$\begin{aligned}\min_{\pi_\theta}\mathbb{E}_{P_{\text{off}}(\boldsymbol{c}|\boldsymbol{x})}\left[\mathbb{D}_{\text{KL}}(P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})||\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}))\right] &\Leftrightarrow \max_{\pi_\theta}\mathbb{E}_{P_{\text{off}}(\boldsymbol{c}|\boldsymbol{x})}\left[\mathbb{E}_{P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})}\left[\log\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})\right]\right]\\ &\Leftrightarrow \max_{\pi_\theta}\mathbb{E}_{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})}\left[\mathbb{E}_{p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})}\left[\log\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})\right]\right],\end{aligned} \tag{1}$$

[1] Conditioning on negative feedback $\boldsymbol{c}^-$ would similarly favor poor responses, though this is rarely useful.

Algorithm 2 Online training: Bootstrapping by conditioning on positive feedback (Section 2.2)

Inputs: Initialize $\theta_0 = \theta_{\text{off}}$ from Algorithm 1, target feedback distribution $p_{\text{user}}(\boldsymbol{c}^+|\boldsymbol{x})$, environment $p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x}, \boldsymbol{o})$, instruct corpus $\mathcal{X}$, training rounds T, steps per round S, batch size B, optimizer $\mathcal{O}$
Outputs: The online-bootstrapped parameters θ_T

for $t = 1$ **to** T **do**
  $\theta_t \leftarrow \theta_{t-1}$
  for all instructions $\boldsymbol{x} \sim \mathcal{X}$ sampled in this round **do**
    Rollout $\boldsymbol{c}^+ \sim p_{\text{user}}(\cdot|\boldsymbol{x})$, $\boldsymbol{o} \sim \pi_{\theta_{t-1}}(\cdot|\boldsymbol{x}, \boldsymbol{c}^+)$ then obtain fresh critique $\boldsymbol{c} \sim p_{\text{env}}(\cdot|\boldsymbol{x}, \boldsymbol{o})$
    Push $(\boldsymbol{x}, \boldsymbol{o}, \boldsymbol{c})$ to buffer $\mathcal{B}^t_{\text{on}}$ // $\boldsymbol{c}$ is usually different (at least linguistically) from $\boldsymbol{c}^+$
  Objective: $\max_{\theta_t} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{o},\boldsymbol{c})\sim\mathcal{B}^t_{\text{on}}}[\log \pi_{\theta_t}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c})]$ // Taking expectation over $\boldsymbol{x} \sim \mathcal{X}$ in Eq. (2)
  for $s = 1$ **to** S **do**
    Sample $\{(\boldsymbol{x}_i, \boldsymbol{o}_i, \boldsymbol{c}_i)\}_{i=1}^B \sim \mathcal{B}^t_{\text{on}}$; $\theta_t \leftarrow \mathcal{O}.\text{step}\big(\theta_t, \nabla_{\theta_t} \frac{1}{B} \sum_{i=1}^B \log \pi_{\theta_t}(\boldsymbol{o}_i|\boldsymbol{x}_i, \boldsymbol{c}_i)\big)$
return θ_T

where the second equivalence follows from the identities $P_{\text{off}}(\boldsymbol{c}|\boldsymbol{x}) \cdot P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}) = P_{\text{off}}(\boldsymbol{o}, \boldsymbol{c}|\boldsymbol{x}) = \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x}, \boldsymbol{o})$. This objective in Eq. (1) reduces to maximum likelihood training. This is exactly what we implement: standard likelihood training on $(\boldsymbol{x}, \boldsymbol{o}, \boldsymbol{c})$ triples with $\boldsymbol{c}$ prepended to the prompt; no reward is computed and no RL objective is optimized (Algorithm 1). Its optimal solution is $\pi^*_\theta(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}) = P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c})$ on the support set of $P_{\text{off}}(\boldsymbol{c}|\boldsymbol{x})$. Notably, our approach does not require explicitly distinguishing positive $\boldsymbol{c}^+$ from negative $\boldsymbol{c}^-$; the language prior embedded in LLMs can implicitly interpret and combine information from diverse forms of feedback, including mixed ones as seen in Figure 1. Further discussion, including the choice of the feedback distribution in Eq. (1), why negative feedback remains useful, and connections to inverse dynamics and goal-conditioned RL, is deferred to Appendix B.

2.2 Online Training: Bootstrapping FCP

We denote the model obtained by solving the offline problem in Eq. (1) as $\pi_{\theta_{\text{off}}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c})$, which is capable of generating responses conditioned on any user-defined feedback $\boldsymbol{c}$. Building on this model, we further perform **online training** to bootstrap performance by *conditioning explicitly on positive feedback* $\boldsymbol{c}^+$. Concretely, we iteratively update parameters θ_{t+1} using rollouts from $\pi_{\theta_t}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$ for $t \in \mathbb{N}$, with $\theta_0 = \theta_{\text{off}}$ initialized from the offline solution, as described in Algorithm 2.

Formally, we define the joint distribution $P_{\theta_t}(\boldsymbol{o}, \boldsymbol{c}, \boldsymbol{c}^+|\boldsymbol{x}) \triangleq p_{\text{user}}(\boldsymbol{c}^+|\boldsymbol{x}) \cdot \pi_{\theta_t}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) \cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x}, \boldsymbol{o})$, where $p_{\text{user}}(\boldsymbol{c}^+|\boldsymbol{x})$ denotes the distribution (fixed or trainable) of user-specified *expected* positive feedback. The corresponding feedback-conditional posterior is

$$P_{\theta_t}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}) = \frac{P_{\theta_t}(\boldsymbol{o}, \boldsymbol{c}|\boldsymbol{x})}{P_{\theta_t}(\boldsymbol{c}|\boldsymbol{x})} = \frac{\sum_{\boldsymbol{c}^+} p_{\text{user}}(\boldsymbol{c}^+|\boldsymbol{x}) \cdot \pi_{\theta_t}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) \cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x}, \boldsymbol{o})}{\sum_{\boldsymbol{o}} \sum_{\boldsymbol{c}^+} p_{\text{user}}(\boldsymbol{c}^+|\boldsymbol{x}) \cdot \pi_{\theta_t}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) \cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x}, \boldsymbol{o})}.$$

The optimization objective for updating θ_{t+1} based on θ_t (with gradients stopped through θ_t) is

$$\begin{aligned} &\min_{\pi_{\theta_{t+1}}} \mathbb{E}_{P_{\theta_t}(\boldsymbol{c}|\boldsymbol{x})} \left[\mathbb{D}_{\text{KL}}(P_{\theta_t}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}) || \pi_{\theta_{t+1}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}))\right] \\ \Leftrightarrow &\max_{\pi_{\theta_{t+1}}} \mathbb{E}_{P_{\theta_t}(\boldsymbol{c}|\boldsymbol{x})} \left[\mathbb{E}_{P_{\theta_t}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})} \left[\log \pi_{\theta_{t+1}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c})\right]\right] \\ \Leftrightarrow &\max_{\pi_{\theta_{t+1}}} \mathbb{E}_{p_{\text{user}}(\boldsymbol{c}^+|\boldsymbol{x})}\Big[\mathbb{E}_{\pi_{\theta_t}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}^+)}\big[\mathbb{E}_{p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})}\big[\log \pi_{\theta_{t+1}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c})\big]\big]\Big] \end{aligned} \tag{2}$$

Intuition. In each training round t (distinct from the s-th gradient steps taken within a round), the current model π_{θ_t} is conditioned on $\boldsymbol{c}^+$ to sample candidate positive responses. These responses are then re-annotated with fresh feedback $\boldsymbol{c}$ from the environment. Over successive rounds, the model learns to identify cases where conditioning on $\boldsymbol{c}^+$ does not in fact yield positive critiques, while reinforcing those that align with the expected feedback. This iterative process bootstraps the model, progressively strengthening alignment with user-specified positive feedback. Moreover, following Lanchantin et al. (2025), the number of gradient steps S between rounds can be flexibly adjusted to interpolate between fully online and semi-online training.

Table 1: **Evaluation results on mathematical benchmarks.** Here **Avg.** denotes mean accuracy (%) over the five math benchmarks, **Env. Model** indicates the environment model used to provide scalar scores and/or verbal feedback, and **RBV** denotes a rule-based verifier.

Offline Algo. + Online Algo.	Env. Model	Scalar Reward	Benchmark AIME24	AIME25	MATH500	Minerva	Olympiad	Avg.
Base	-	-	7.5 ±1.7	6.7 ±0.0	63.8 ±0.4	28.3 ±0.8	28.6 ±0.4	27.0 ±0.5
+ GRPO	RBV	✓	13.3 ±2.7	14.2 ±4.2	76.3 ±0.6	36.6 ±0.4	**41.7** ±0.4	36.4 ±0.7
+ GRPO	LLM	✓	20.0 ±0.0	13.3 ±2.7	75.7 ±1.7	42.3 ±1.3	40.8 ±0.5	38.4 ±0.9
+ Critique-GRPO	LLM+RBV	✓	15.0 ±1.9	9.2 ±1.7	76.8 ±0.3	36.1 ±0.9	40.1 ±0.5	35.4 ±0.6
RFT	RBV	✓	13.3 ±0.0	3.3 ±0.0	69.2 ±0.6	32.4 ±0.6	33.8 ±0.9	30.4 ±0.0
+ GRPO	RBV	✓	17.5 ±1.7	**15.0** ±1.9	**77.0** ±0.8	38.4 ±0.7	41.3 ±0.7	37.8 ±0.4
+ GRPO	LLM	✓	**25.8** ±1.7	9.2 ±1.7	75.1 ±0.7	36.8 ±0.9	38.9 ±0.1	37.1 ±0.5
+ Critique-GRPO	LLM+RBV	✓	16.7 ±4.7	9.2 ±5.0	75.2 ±0.4	35.8 ±0.7	39.6 ±0.5	35.3 ±1.2
CFT	LLM	✗	1.7 ±3.3	0.0 ±0.0	27.0 ±3.2	9.2 ±6.4	7.7 ±1.4	9.1 ±1.3
FCP	LLM	✗	6.7 ±0.0	3.3 ±3.8	68.9 ±1.0	31.2 ±1.1	32.4 ±1.1	28.5 ±1.1
+ Bootstrap	LLM	✗	25.0 ±3.3	7.5 ±1.7	76.5 ±0.7	**45.8** ±0.7	38.8 ±0.6	**38.7** ±0.7

Table 2: **General reasoning results.** Accuracy (%) across three benchmarks and their average. For MMLU-Pro, we report results from a single evaluation run, since its large test set (∼12K questions) makes evaluation costly while ensuring low variance (typically ≤ 0.3).

Offline Algo. + Online Algo.	Env. Model	Scalar Reward	GPQA-Diamond	MMLU-Pro	TheoremQA	Average
Base	-	-	27.9 ±1.0	49.7	38.6 ±0.2	38.7 ±0.4
+ GRPO	LLM	✓	32.5 ±5.3	49.7	**49.4** ±1.4	43.9 ±1.7
RFT	RBV	✓	35.2 ±1.3	55.0	43.7 ±0.9	44.6 ±0.2
+ GRPO	LLM	✓	37.2 ±2.5	**57.0**	48.3 ±0.2	47.5 ±0.8
FCP	LLM	✗	35.0 ±2.9	53.6	42.0 ±1.0	43.5 ±0.6
+ Bootstrap	LLM	✗	**39.1** ±2.9	55.3	49.1 ±0.5	**47.8** ±0.9

Test-time usage. At test time, FCP does not require any environment-generated feedback. If the user wishes, they may provide a desired feedback condition $\boldsymbol{c}_{\text{user}}$. When no such condition is given, we can automatically obtain a positive feedback description by prompting an LLM with instructions such as: "`Provide several possible positive feedback descriptions for the following query.`" This produces a suitable $\boldsymbol{c}^+$ for conditioning. The FCP model then returns an output $\boldsymbol{o}$ from $\pi_\theta(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$. This is a single-turn generation process that does not require any iterative refinement. In practice, inference simply consists of prepending the selected feedback description to the input prompt.

3 Experiments

We evaluate FCP across complex reasoning and open-ended creative writing tasks, and further examine its behavioral controllability under diverse feedback conditions.

3.1 Setup

Tasks and feedback. Reasoning experiments use `Qwen2.5-7B-Base` (Yang et al., 2024a), trained on `Big-Math` (Albalak et al., 2025) for mathematics and `WebInstruct` (Yue et al., 2024; Ma et al., 2025) for general reasoning. `GPT-5-nano` supplies both verbal critiques and scalar scores through a unified prompt (Figure 5). Its reviewer-style critiques cover multiple response attributes; user-style critiques provide a less detailed alternative. Creative writing uses a separate environment and evaluator, described in Section 3.3. All training feedback is model-generated.

Baselines. RFT (Dong et al., 2023) finetunes on responses selected for correctness by a rule-based verifier (RBV). GRPO (Shao et al., 2024) is trained from either Base or RFT checkpoint using either RBV rewards or LLM-assigned scalar scores. Critique-GRPO (Zhang et al., 2025) adds critique-guided refinements to scalar-reward training, and CFT (Wang et al., 2025a) imitates critiques as targets.

Training details for FCP. In the *offline* stage (Algorithm 1), the base model generates 8 responses per prompt. We use `GPT-5-nano` both to provide feedback for each response and to classify the resulting feedback. All collected feedback is used to train FCP, while the classification results are used to construct a pool of positive conditions $\{\boldsymbol{c}^+\}$ for online conditioning.

In the *online* stage (Algorithm 2), for each prompt $\boldsymbol{x}$ we sample a desired condition $\boldsymbol{c}^+ \sim p_{\text{user}}(\cdot|\boldsymbol{x})$ by drawing from the pool $\{\boldsymbol{c}^+\}$. For rollout, the prompt batch size is 2048 with 4 responses per prompt; for training, the mini-batch size is 512, giving 4 gradient updates per rollout step. Each response receives fresh feedback from `GPT-5-nano`, which is concatenated with the prompt and response for training (using the Algorithm 1 wrapper `<EF>` and `</EF>`). This bootstrapping loop improves response quality under desired conditions while grounding updates in new feedback. GRPO uses scalar scores from the same feedback environment. Hyperparameters, implementations, and measured costs are in Appendix D.

Evaluation. We assess mathematical reasoning on AIME24&25, MATH500 (Hendrycks et al., 2021), Minerva-Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024), and general reasoning on GPQA-Diamond (Rein et al., 2024), MMLU-Pro (Wang et al., 2024), and TheoremQA (Chen et al., 2023). All benchmarks use a unified protocol: each dataset is run under four random seeds, with mean accuracy reported. Inference uses `vllm` (Kwon et al., 2023) with greedy decoding and a maximum generation length of 8192 tokens. For FCP, we match the training setup by randomly sampling one feedback condition from $\{\boldsymbol{c}^+\}$ for each question and prepending it to the prompt template.

3.2 Reasoning

Offline FCP is a modest start. Offline FCP reaches 28.5% average math accuracy, between Base (27.0%) and RFT (30.4%); general reasoning shows the same order, with Base at 38.7%, FCP at 43.5%, and RFT at 44.6% (Tables 1 and 2). Offline conditioning thus improves on Base but not on RFT, consistent with the posterior view when the positive condition mostly encodes correctness (Section 2.1).

Bootstrapping reaches GRPO-level accuracy. Online training raises FCP to 38.7% on math. The four GRPO baselines span 36.4–38.4%: from Base, 38.4% with LLM rewards and 36.4% with RBV; from RFT, 37.1% with LLM rewards and 37.8% with RBV. On general reasoning, FCP reaches 47.8%, against 47.5% for RFT+GRPO and 43.9% for Base+GRPO. FCP+Bootstrap is therefore level with the strongest scalar-reward baseline in each suite. This pattern largely holds on two additional backbones, `Qwen3-4B-Base` and `Llama-3.1-8B`, where FCP+Bootstrap is within 1.6 points of the best GRPO baseline (Table 10). Two critique-based alternatives fare worse in our setup: Critique-GRPO reaches 35.4% from Base and 35.3% from RFT, and critique imitation (CFT) 9.1%.

3.3 Open-Ended Creative Writing

While mathematical and general reasoning benefit from verifiable correctness, open-ended creative writing poses an intrinsically non-verifiable challenge where response quality is multi-dimensional and poorly captured by a single scalar. To evaluate FCP in this regime, we train offline on creative-writing response–feedback pairs from LitBench (Fein et al., 2026), starting from `Qwen2.5-7B-Base`, and perform online bootstrapping from this checkpoint. The environment model, `Qwen3.5-122B-A10B`, critiques each rollout via a structured prompt (Figure 6) across four distinct dimensions: narrative execution, characterization and emotional impact, originality and creativity, and prose and style. For each dimension that is satisfactorily addressed, the environment model assigns `perfect`. Rather than sampling from a feedback pool as in reasoning, we set $\boldsymbol{c}^+$ to a fixed condition specifying `perfect` across all four dimensions for rollouts and inference, while policy updates continue to train on the diverse textual critiques.

All checkpoints are evaluated by LitBench's independent, human-preference-aligned reward model[2] (Table 3). While offline training provides only marginal gains over Base (53.0% win rate for FCP vs. 48.2% for RFT), online bootstrapping establishes a decisive advantage for FCP over scalar-reward optimization. Specifically, FCP+Bootstrap attains the highest reward (-0.31, compared to -0.71 for Base+GRPO and -0.59 for RFT+GRPO) and a 91.7% win rate against Base. Crucially, in head-to-head comparisons, the independent evaluator strongly prefers FCP+Bootstrap over Base+GRPO (68.0%) and RFT+GRPO (62.7%). Because both GRPO baselines are trained on scalar rewards

[2] https://huggingface.co/SAA-Lab/Llama8B-CreativeWritingVerifier

Table 3: **Creative writing results on LitBench.** Higher is better. FCP+Bootstrap substantially outperforms both offline and online scalar-reward baselines in reward and direct head-to-head win rates. All metrics are evaluated by LitBench's held-out reward model. The lower panel reports direct win rates against online baselines.

Overall evaluation		
Method	Win vs. Base (%)	Reward
Base	50.0	-1.74 $\pm$0.01
Base + GRPO	88.1	-0.71 $\pm$0.01
RFT	48.2	-1.82 $\pm$0.01
RFT + GRPO	90.6	-0.59 $\pm$0.02
FCP	53.0	-1.68 $\pm$0.02
FCP + Bootstrap	**91.7**	**-0.31** $\pm$0.01
Direct comparison against FCP + Bootstrap		
Opponent	FCP + Bootstrap win (%)	
Base + GRPO	68.0	
RFT + GRPO	62.7	

Table 4: **Comparison under different feedback conditions.** Accuracy (%), code ratio (proportion of responses containing code), and average response length are all measured on MATH500. For the Instruct model, evaluation prompts are wrapped as "`Your answer should be expected to get the following critique: <feedback_content>\n{question}`".

		Example 1	Example 2	Example 3	Example 4
Feedback type		`fully_positive`	`fully_negative`	`neutral`	`has_code`
Content		Accurate and clear; concise and coherent reasoning; correct conclusion.	Incoherent and **incomplete**. Random and unfocused. Unclear and disorganized.	Correct and readable overall, but the solution is **verbose** and could be streamlined for tighter logical flow.	Correct and clear, though slightly verbose with superfluous **code**.
Accuracy	Instruct	76.2	77.4	77.5	76.6
	FCP	68.5	**17.1**	61.1	53.9
Code Ratio	Instruct	0	0	0	0
	FCP	22.7	55.6	46.3	**74.3**
Response Length	Instruct	632	650	638	661
	FCP	605	1442	**722**	659

computed by averaging the scores across all four dimensions from the exact same environment model, this controlled comparison suggests that retaining multi-dimensional verbal critiques mitigates the information bottleneck inherent to scalarization, allowing the policy to synthesize higher-quality open-ended text.

3.4 Feedback-Conditioned Controllability

FCP enables controllable behavior across diverse feedback conditions. A core question is whether the policy truly *learns* the conditioning signal $\boldsymbol{c}$—and, if so, whether this lets us absorb negative samples into training without hurting best-case performance. We probe this by sampling representative feedback from the offline pool and evaluating under several conditions.

Table 4 shows a sharp contrast on MATH500: accuracy is 68.5% under `fully_positive` but only 17.1% under `fully_negative`, far below the base model's 63.8% (Table 1). This indicates the model internalizes the control signal: negative conditions induce poor behavior when requested, yet positive conditions still yield strong accuracy—showing that including negative samples in training does not cap performance under positive conditions.

Other conditions also shift behavior as intended. Under `neutral`, where the condition $\boldsymbol{c}$ asks for a correct answer and a more verbose solution, accuracy drops but response length grows. With `has_code`, the share of responses containing code rises to 74.3%, confirming that stylistic attributes in $\boldsymbol{c}$ are also followed. Compared to `Qwen2.5-7B-Instruct`, which shows little variation across

Table 5: Training feedback variants on math benchmarks. Variants change content, style, source, or reliability; they do not form a controlled information scale. All User feedback is model-generated.

Method	Feedback Style	Variant	AIME24	AIME25	MATH500	Minerva	Olympiad	Avg.
Base	-	-	7.5 ±1.7	6.7 ±0.0	63.8 ±0.4	28.3 ±0.8	28.6 ±0.4	27.0 ±0.5
FCP + Bootstrap	Correctness-Only	Binary labels	10.0 ±2.7	5.0 ±1.9	73.4 ±0.7	34.3 ±0.6	35.3 ±0.4	31.6 ±0.8
FCP + Bootstrap	User	Colloquial	16.7 ±2.7	0.8 ±1.7	72.2 ±0.4	37.1 ±0.4	37.1 ±0.8	32.8 ±0.8
FCP + Bootstrap	Reviewer-Noisy	10% shuffled	15.8 ±1.7	4.2 ±1.7	75.3 ±0.6	36.1 ±0.8	36.2 ±0.7	33.5 ±0.2
FCP + Bootstrap	Reviewer-Lite	Short attributes	14.2 ±4.2	8.3 ±1.9	74.0 ±0.3	37.4 ±1.5	37.3 ±0.7	34.2 ±1.0
FCP + Bootstrap	Reviewer	Full critique	25.0 ±3.3	7.5 ±1.7	76.5 ±0.7	45.8 ±0.7	38.8 ±0.6	38.7 ±0.7

Table 6: **Feedback-conditioning controls.** Five-task mean math accuracy (%). Base is 27.0%. Online no-feedback reports its best checkpoint at step 5. Scalar uses a fixed target score; verbal uses a positive-condition pool.

Condition	Offline	Online	Online gain (points)
None	28.1	28.9	+0.8
Textualized scalar	29.8	33.6	+3.8
Verbal FCP (reviewer-style)	28.5	38.7	+10.2

conditions due to training only on verified positives, FCP learns to map feedback $\boldsymbol{c}$ to distinct behaviors, enabling broad data use without manual filtering.

4 Ablation Studies

In this section, we conduct three ablations: (i) varying the quality of verbal feedback during training, (ii) comparing against scalar conditioning and no-feedback self-training, and (iii) testing policy sensitivity to a weaker critique model. Additional ablation results and analyses are deferred to Appendix C.

4.1 Effect of Feedback Content during Training

Since FCP conditions directly on textual feedback, its performance may depend on the informativeness and reliability of the training feedback. We therefore compare five feedback configurations spanning different quality levels: **Correctness-Only** provides only "correct" or "incorrect" labels using `math-verify`;[3] **User** prompts `GPT-5-nano` to simulate colloquial and subjective user comments following Table 8; **Reviewer-Noisy** shuffles 10% of online feedback to introduce mismatched supervision; **Reviewer-Lite** uses short attribute-level critiques generated by `GPT-5-nano`, e.g., "correct, clear, concise, straightforward"; and **Reviewer** uses high-quality free-form critiques as in Table 8.

Table 5 shows that average math accuracy increases with feedback quality. FCP+Bootstrap improves from 31.6% with **Correctness-Only** feedback to 32.8% with simulated **User** feedback, 34.2% with **Reviewer-Lite** feedback, and 38.7% with **Reviewer** feedback. This trend suggests that richer textual critiques provide a more informative training signal than binary correctness labels, as they capture multiple dimensions of response quality, such as clarity, structure, and completeness.

The **Reviewer-Noisy** setting shows that FCP+Bootstrap is sensitive to feedback–response mismatch: shuffling 10% of online feedback lowers accuracy from 38.7% to 33.5%, though it remains well above the 27.0% base model.

4.2 Scalar Conditioning and No-Feedback Training

To rigorously understand the source of FCP's gains, we disentangle two critical factors: (i) whether the online improvement stems from external feedback or merely from iterative self-training rollouts, and (ii) whether rich verbal descriptions offer benefits beyond simple scalar rewards. To this end, we benchmark FCP against two controlled variants: **No-Feedback**, which removes feedback conditions from both offline data and online rollouts (reducing to standard online self-training), and **Textualized-Scalar** (inspired by Yang et al. (2024b)), which replaces free-form critiques with discretized scores (e.g., `<EF>Score:7/10</EF>`) and conditions rollouts on high scores (`Score:9/10`).

[3] https://github.com/huggingface/Math-Verify

Table 7: Comparison of FCP performance when trained with weaker critique models and different feedback styles. `Qwen2.5-1.5B-Instruct` serves as a substantially weaker critique model to test whether FCP relies on a strong feedback provider.

Model / Method	Critique Model	Feedback Style	AIME24	AIME25	MATH500	Minerva	Olympiad	Avg.
Qwen2.5-1.5B-Instruct	-	-	0.0 ±0.0	0.0 ±0.0	55.0 ±1.3	20.5 ±0.6	20.3 ±0.6	19.2 ±0.3
Qwen2.5-7B-Base	-	-	7.5 ±1.7	6.7 ±0.0	63.8 ±0.4	28.3 ±0.8	28.6 ±0.4	27.0 ±0.5
w/ FCP + Bootstrap	Qwen2.5-1.5B-Instruct	User	14.2 ±5.7	5.8 ±1.7	72.9 ±0.9	35.5 ±0.7	37.4 ±0.6	33.1 ±1.4
w/ FCP + Bootstrap	GPT-5-nano	User	16.7 ±2.7	0.8 ±1.7	72.2 ±0.4	37.1 ±0.4	37.1 ±0.8	32.8 ±0.8
w/ FCP + Bootstrap	GPT-5-nano	Reviewer	25.0 ±3.3	7.5 ±1.7	76.5 ±0.7	45.8 ±0.7	38.8 ±0.6	38.7 ±0.7

Table 6 reveals distinct dynamics across these paradigms. In the offline phase, all three variants perform similarly (28.1%–29.8%), indicating that simply conditioning on different formats without interaction provides limited differentiation. However, their trajectories diverge sharply during online bootstrapping. The **No-Feedback** variant stagnates with a marginal +0.8% gain (peaking at 28.9% before degrading), demonstrating that naive self-training without grounding feedback cannot sustain policy improvement. While **Textualized-Scalar** achieves steady progress (+3.8% online gain to 33.6%), full verbal FCP achieves a substantial +10.2% boost to 38.7%. These results confirm that environment feedback is indispensable for online bootstrapping, and that, when used as conditions, natural-language critiques provide a substantially stronger learning signal than textualized scalar scores.

4.3 Weaker Critique Model

One potential concern is that FCP might benefit from a spurious advantage by using verbal critiques generated by `GPT-5-nano`, rather than from the FCP training framework itself. To isolate this effect, we replace the critique model with a substantially smaller and weaker model, `Qwen2.5-1.5B-Instruct`. As shown in Table 7, using this model as the provider of user-style feedback yields an average accuracy of 33.1% on the math suite, comparable to the 32.8% obtained with `GPT-5-nano` and well above the base model's 27.0%, indicating that FCP does not rely on a strong critique model to achieve meaningful gains. Notably, `Qwen2.5-1.5B-Instruct` itself is weaker on these benchmarks (average accuracy 19.2% vs. 27.0% for `Qwen2.5-7B-Base`), arguing against the explanation that FCP is simply distilling knowledge from a stronger teacher.

5 Discussion and Future Directions

Our key insight is that the essence of RL lies in *online interaction with the environment*, not in scalar rewards or any specific algorithm. Scalarization was historically necessary for control-centric RL in robotics or strategy-centric RL in games, but it may not be intrinsic to language-centric systems like LLMs. This reopens the debate around the reward hypothesis: earlier critics could only offer counterexamples without an alternative framework (Skalse & Abate, 2022), whereas our FCP approach leverages **language priors** to provide a principled way to bypass scalar rewards. Crucially, during training, feedback $\boldsymbol{c}$ is a *dependent variable* generated from the environment $p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x}, \boldsymbol{o})$ and cannot be directly controlled, while at test time the conditioning feedback $\boldsymbol{c}^+$ becomes an *independent variable* freely specified by users. This asymmetry enables full use of diverse feedback during training while allowing controllability at inference. By directly mapping feedback to responses, FCP preserves the richness of verbal feedback that scalarization discards (Sections 3.3 and 4.2). Unlike RFT (Dong et al., 2023; Touvron et al., 2023), which discards many useful data pairs, FCP retains diverse feedback, including mixed and uncertain, and can merge complementary signals across examples at test time (Figure 1). This establishes verbal feedback as a first-class training signal and FCP as a natural, scalable alternative to scalarized RL.

Future directions. Several extensions of FCP are promising. One is to *combine it with verifiable rewards*, for instance by treating the absence of feedback as a neutral condition (e.g., using the null feedback token `<EF></EF>`), so that reliable scalar supervision can complement verbal feedback when available. Another is to extend FCP to *multi-turn interactions*, where feedback is incorporated before the next turn of generation in a teacher-forcing style, enabling closer alignment with iterative human guidance. A third is *test-time adaptation*: by conditioning on a few user-provided examples, the model could rapidly adjust to individual feedback styles, similar to personalization in text-to-image generation. Finally, the feedback condition $\boldsymbol{c}$ could be made *multimodal*. Collectively, these future directions would deepen integration of natural feedback into LLM training, bridging offline and online stages while adapting to diverse user needs.

AI USE STATEMENT

We used an OpenAI LLM (GPT-5 Series) as a writing and formatting assistant. In particular, it helped refine grammar and phrasing, improve clarity, and suggest edits to figure/table captions and layout (e.g., column alignment, caption length, placement). The LLM did not contribute to research ideation, experimental design, implementation, data analysis, or technical content beyond surface-level edits. All outputs were reviewed and edited by the authors, who take full responsibility for the final text and visuals.

REPRODUCIBILITY STATEMENT

The FCP training procedure is specified in Algorithms 1 and 2 and Section 3.1. Hyperparameters are listed in Table 13, compute costs in Table 14, the prompt templates used to elicit feedback in Figures 5 and 6, and examples of positive conditions in Table 15. Derivations are given in Appendix B. All datasets and base models are publicly available (Appendix D).

REFERENCES

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models, 2025. URL https://arxiv.org/abs/2502.17387.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Zachary Ankner, Mansheej Paul, Brandon Cui, Jonathan D Chang, and Prithviraj Ammanabrolu. Critique-out-loud reward models. *arXiv preprint arXiv:2408.11791*, 2024.

Sanjeev Arora, Simon Du, Sham Kakade, Yuping Luo, and Nikunj Saunshi. Provable representation learning for imitation learning via bi-level optimization. In *International Conference on Machine Learning*, pp. 367–376. PMLR, 2020.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

David Brandfonbrener, Ofir Nachum, and Joan Bruna. Inverse dynamics pretraining learns good representations for multitask imitation. *Advances in Neural Information Processing Systems*, 36: 66953–66978, 2023.

Angelica Chen, Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Samuel R. Bowman, Kyunghyun Cho, and Ethan Perez. Learning from natural language feedback. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. Theoremqa: A theorem-driven question answering dataset. *arXiv preprint arXiv:2305.12524*, 2023.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Acereason-nemotron: Advancing math and code reasoning through reinforcement learning. *arXiv preprint arXiv:2505.16400*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

Yonathan Efroni, Dipendra Misra, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Provable rl with exogenous distractors via multistep inverse dynamics. *arXiv preprint arXiv:2110.08847*, 2021.

Daniel Fein, Sebastian Russo, Violet Xiang, Kabir Jolly, Rafael Rafailov, and Nick Haber. Litbench: A benchmark and dataset for reliable evaluation of creative writing. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7740–7755, 2026.

Jiaxuan Gao, Shusheng Xu, Wenjie Ye, Weilin Liu, Chuyi He, Wei Fu, Zhiyu Mei, Guangju Wang, and Yi Wu. On designing effective rl reward at training time for llm reasoning. *arXiv preprint arXiv:2410.15115*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, et al. Openthoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*, 2025.

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

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3), 2018.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems, 2024.

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

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *arXiv preprint arXiv:2504.07086*, 2025.

Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling reinforcement learning on the base model. https://github.com/Open-Reasoner-Zero/Open-Reasoner-Zero, 2025.

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

Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, Tianyu Pang, and Wenhu Chen. Verltool: Towards holistic agentic reinforcement learning with tool use. *arXiv preprint arXiv:2509.01055*, 2025.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025a.

Chuanyang Jin, Jing Xu, Bo Liu, Leitian Tao, Olga Golovneva, Tianmin Shu, Wenting Zhao, Xian Li, and Jason Weston. The era of real-world human interaction: Rl from user conversations. *arXiv preprint arXiv:2509.25137*, 2025b.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Aviral Kumar, Xue Bin Peng, and Sergey Levine. Reward-conditioned policies. *arXiv preprint arXiv:1912.13465*, 2019.

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

Jack Lanchantin, Angelica Chen, Janice Lan, Xian Li, Swarnadeep Saha, Tianlu Wang, Jing Xu, Ping Yu, Weizhe Yuan, Jason E Weston, et al. Bridging offline and online reinforcement learning for llms. *arXiv preprint arXiv:2506.21495*, 2025.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiawei Liu and Lingming Zhang. Code-r1: Reproducing r1 for code with reliable rewards. https://github.com/ganler/code-r1, 2025.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025a.

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling. *arXiv preprint arXiv:2504.02495*, 2025b.

Saüc Abadal Lloret, Shehzaad Dhuliawala, Keerthiram Murugesan, and Mrinmaya Sachan. Towards aligning language models with textual feedback. *arXiv preprint arXiv:2407.16970*, 2024.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl, 2025. Notion Blog.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. General-reasoner: Advancing llm reasoning across all domains. *arXiv preprint arXiv:2505.14652*, 2025.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative reward models. *arXiv preprint arXiv:2410.12832*, 2024.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pp. 745–750, 2007.

Penghui Qi, Zichen Liu, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Optimizing anytime reasoning via budget relative policy optimization. *arXiv preprint arXiv:2505.13438*, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

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

Alireza Salemi and Hamed Zamani. Learning from natural language feedback for personalized question answering. *arXiv preprint arXiv:2508.10695*, 2025.

Max Schwarzer, Nitarshan Rajkumar, Michael Noukhovitch, Ankesh Anand, Laurent Charlin, R Devon Hjelm, Philip Bachman, and Aaron C Courville. Pretraining representations for data-efficient reinforcement learning. *Advances in Neural Information Processing Systems*, 34:12686–12699, 2021.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Taiwei Shi, Zhuoer Wang, Longqi Yang, Ying-Chun Lin, Zexue He, Mengting Wan, Pei Zhou, Sujay Kumar Jauhar, Sihao Chen, Shan Xia, et al. Wildfeedback: Aligning llms with in-situ user interactions and feedback. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 36701–36725, 2026.

Joar Max Viktor Skalse and Alessandro Abate. The reward hypothesis is false. In *NeurIPS ML Safety Workshop*, 2022.

Moritz Stephan, Alexander Khazatsky, Eric Mitchell, Annie S Chen, Sheryl Hsu, Archit Sharma, and Chelsea Finn. Rlvf: Learning from verbal feedback without overgeneralization. *arXiv preprint arXiv:2402.10893*, 2024.

Richard Sutton. The reward hypothesis. http://incompleteideas.net/rlai.cs.ualberta.ca/RLAI/rewardhypothesis.html, 2004.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Yubo Wang, Xiang Yue, and Wenhu Chen. Critique fine-tuning: Learning to critique is more effective than learning to imitate. *arXiv preprint arXiv:2501.17703*, 2025a.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, et al. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025b.

Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, Lingming Zhang, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida I Wang. Swe-rl: Advancing llm reasoning via reinforcement learning on open software evolution. *arXiv preprint arXiv:2502.18449*, 2025.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, et al. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*, 2025.

Chenxi Whitehouse, Tianlu Wang, Ping Yu, Xian Li, Jason Weston, Ilia Kulikov, and Swarnadeep Saha. J1: Incentivizing thinking in llm-as-a-judge via reinforcement learning. *arXiv preprint arXiv:2505.10320*, 2025.

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

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

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

John Yang, Kilian Lieret, Carlos E Jimenez, Alexander Wettig, Kabir Khandpur, Yanzhe Zhang, Binyuan Hui, Ofir Press, Ludwig Schmidt, and Diyi Yang. Swe-smith: Scaling data for software engineering agents. *arXiv preprint arXiv:2504.21798*, 2025b.

Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. In *International Conference on Machine Learning*, pp. 56276–56297. PMLR, 2024b.

Shunyu Yao. The second half. https://ysymyth.github.io/The-Second-Half/, 2025.

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

Xiang Yue, Tianyu Zheng, Ge Zhang, and Wenhu Chen. Mammoth2: Scaling instructions from the web. *Advances in Neural Information Processing Systems*, 37:90629–90660, 2024.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Hongyu Zang, Xin Li, Jie Yu, Chen Liu, Riashat Islam, Remi Tachet Des Combes, and Romain Laroche. Behavior prior representation learning for offline reinforcement learning. *arXiv preprint arXiv:2211.00863*, 2022.

Weihao Zeng, Yuzhen Huang, Wei Liu, Keqing He, Qian Liu, Zejun Ma, and Junxian He. 7b model and 8k examples: Emerging reasoning with reinforcement learning is both effective and efficient. https://hkust-nlp.notion.site/simplerl-reason, 2025. Notion Blog.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*, 2024.

Xiaoying Zhang, Hao Sun, Yipeng Zhang, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen Meng. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback. *arXiv preprint arXiv:2506.03106*, 2025.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatgpt interaction logs in the wild. *arXiv preprint arXiv:2405.01470*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P Xing, et al. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *arXiv preprint arXiv:2309.11998*, 2023.

Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *international conference on machine learning*, pp. 27042–27059. PMLR, 2022.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang, Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin, and Chao Du. Reinforcing general reasoning without verifiers. *arXiv preprint arXiv:2505.21493*, 2025.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

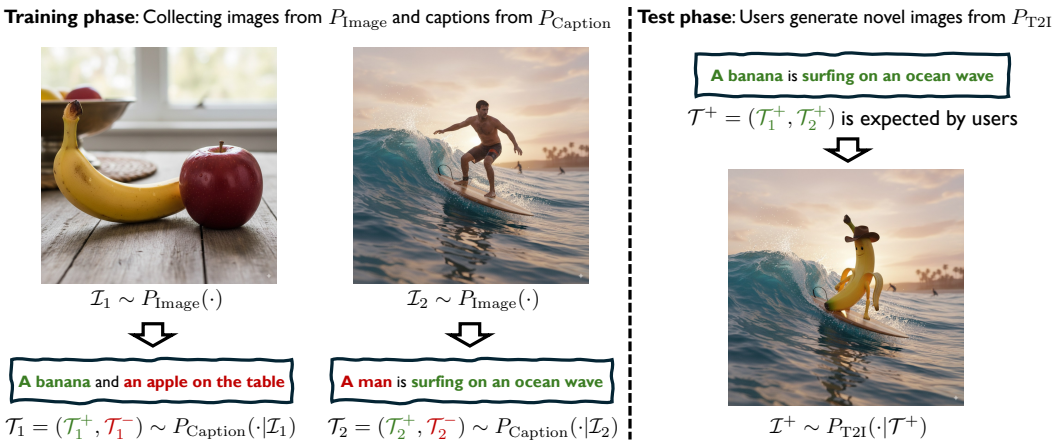


Figure 2: **Learning from mixed captions in text-to-image generation.** During training, models learn from realistic image-caption pairs such as "*a banana and an apple on the table*" or "*a man surfing on an ocean wave*". They can leverage language priors to recombine these captions and generate novel concepts, such as "*a banana surfing on the ocean*" (images shown are generated with Gemini 2.5 Flash Image). By analogy to Figure 1, this illustrates how diverse verbal feedback can be treated as a conditioning signal, motivating our feedback-conditional learning paradigm.

A Related Work

SFT and RL methods for reasoning. The ability to perform reasoning has become a defining strength of LLMs, enabling progress across mathematics, coding, and scientific domains (Jaech et al., 2024; Comanici et al., 2025). To enhance these skills, two approaches have proven especially influential: SFT and RL (Uesato et al., 2022; Rafailov et al., 2023; Guha et al., 2025; Hu et al., 2025; Hochlehnert et al., 2025). Following the success of the DeepSeek-R1 recipe (Shao et al., 2024; Guo et al., 2025), a number of RL variants have been introduced, including Dr. GRPO (Liu et al., 2025a), DAPO (Yu et al., 2025), REINFORCE++ (Hu, 2025), and VAPO (Yue et al., 2025). Beyond algorithmic proposals, researchers have systematically investigated the RL design space for reasoning (Zeng et al., 2025; Team et al., 2025), examining factors such as staged training curricula (Wen et al., 2025; Luo et al., 2025) and reward formulation (Gao et al., 2024; Cui et al., 2025; Qi et al., 2025; Zhou et al., 2025). While much of the initial progress focused on mathematics, these methods have more recently been extended to software engineering and code reasoning (Liu & Zhang, 2025; Xie et al., 2025; Wei et al., 2025; Yang et al., 2025b; Chen et al., 2025), as well as to broader agentic applications (Wang et al., 2025b; Jin et al., 2025a; Jiang et al., 2025; Xue et al., 2025).

Learning from verbal feedback. Most existing approaches convert verbal feedback into scalar rewards for RL training (Kim et al., 2024; Ankner et al., 2024; Lightman et al., 2024; Stephan et al., 2024; Whitehouse et al., 2025; Liu et al., 2025b). More recent efforts explore learning directly from feedback or critiques: Lloret et al. (2024) propose conditional SFT based on toxicity categorization in alignment tasks, CFT (Wang et al., 2025a) trains models to imitate critiques, Critique-GRPO (Zhang et al., 2025) incorporates critique-guided refinements into online RL, Salemi & Zamani (2025) jointly optimize a feedback model and a policy model, and Chen et al. (2024) introduce a refinement model that corrects errors using feedback. These approaches generally assume feedback is high-quality, informative or categorized, and reliably improves self-refinement. In practice, however, verbal feedback, even from strong LLM critics, is often mixed across criteria, loosely structured, or hedged, and this is likely to be more pronounced for feedback from human users. In contrast, our FCP framework does not require feedback to be high-quality or rubric-constrained; by treating feedback as a conditioning signal rather than a prediction target, it can flexibly exploit the full range of verbal feedback, including noisy or mixed forms, for training. A complementary line of work learns from in-the-wild user interactions by converting user follow-ups into preferences, rewrites, or rewards (Shi et al., 2026; Jin et al., 2025b); applying FCP directly to such naturally occurring feedback is a promising direction we leave to future work.

B Additional Derivations and Discussions

B.1 Connection to KL-Regularized RL

This section makes precise the connection mentioned in Section 2.1: the feedback-conditional posterior $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$ is the optimal solution to a KL-constrained reward maximization problem with reward function $\log p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})$,

$$P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) \in \arg\max_{\pi} \mathbb{E}_{\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)} \left[\log p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})\right] - \mathbb{D}_{\text{KL}}\left(\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) \,\|\, \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})\right). \tag{3}$$

FCP does *not* optimize Eq. (3); this connection only serves to interpret the distribution that FCP targets.

Proof of Eq. (3). Following Rafailov et al. (2023), the optimal solution to the objective in Eq. (3) can be written as

$$\begin{aligned}
\pi^*(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) &= \frac{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot \exp\left(\log p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})\right)}{\sum_{\boldsymbol{o}} \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot \exp\left(\log p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})\right)} \\
&= \frac{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})}{\sum_{\boldsymbol{o}} \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})} \\
&= \frac{P_{\text{off}}(\boldsymbol{o}, \boldsymbol{c}^+|\boldsymbol{x})}{P_{\text{off}}(\boldsymbol{c}^+|\boldsymbol{x})} = P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+).
\end{aligned} \tag{4}$$

Moreover, the objective in Eq. (3) is equivalent to minimizing the *reverse* KL divergence between $\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$ and $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$:

$$\begin{aligned}
&\mathbb{E}_{\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)}\left[\log p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})\right] - \mathbb{D}_{\text{KL}}\left(\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) || \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})\right) \\
=& -\mathbb{D}_{\text{KL}}\left(\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) || P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)\right) + \log P_{\text{off}}(\boldsymbol{c}^+|\boldsymbol{x}).
\end{aligned} \tag{5}$$

Why FCP does not optimize Eq. (3). Both the reward in Eq. (3) and the reverse KL divergence in Eq. (5) require evaluating $\log p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})$ for responses sampled from π. When feedback comes from black-box LLM critics or human users, we can only sample from p_{env} and cannot evaluate this likelihood. FCP instead minimizes the *forward* KL divergence in Eq. (1), which only requires samples from π_{ref} and p_{env}. Over an unrestricted policy class, the forward and reverse KL divergences share the same minimizer $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$; for a finite-capacity model, their solutions generally differ, as the forward KL divergence is mass-covering whereas the reverse KL divergence is mode-seeking.

Special case: verifiable rewards. In the special case where the environment provides *verifiable rewards*, that is, $p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o}^+) = 1$ for correct responses $\boldsymbol{o}^+$ and $p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o}^-) = 0$ for incorrect responses $\boldsymbol{o}^-$, we can show that $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$ reduces to the optimal solution of a 0-1 reward maximization problem without KL regularization: $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) \in \arg\max_{\pi} \mathbb{E}_{\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)}\left[\mathbb{1}(\boldsymbol{o} \text{ is } \boldsymbol{o}^+)\right]$. Specifically, we have

$$\begin{aligned}
P_{\text{off}}(\boldsymbol{o}^+|\boldsymbol{x}, \boldsymbol{c}^+) &= \frac{\pi_{\text{ref}}(\boldsymbol{o}^+|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o}^+)}{\sum_{\boldsymbol{o}} \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})} = \frac{\pi_{\text{ref}}(\boldsymbol{o}^+|\boldsymbol{x})}{\sum_{\boldsymbol{o} \text{ is } \boldsymbol{o}^+} \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})}; \\
P_{\text{off}}(\boldsymbol{o}^-|\boldsymbol{x}, \boldsymbol{c}^+) &= \frac{\pi_{\text{ref}}(\boldsymbol{o}^-|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o}^-)}{\sum_{\boldsymbol{o}} \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x}, \boldsymbol{o})} = 0.
\end{aligned} \tag{6}$$

Thus, substituting $\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) = P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$ into $\mathbb{E}_{\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)}\left[\mathbb{1}(\boldsymbol{o} \text{ is } \boldsymbol{o}^+)\right]$, we have

$$\mathbb{E}_{P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)}\left[\mathbb{1}(\boldsymbol{o} \text{ is } \boldsymbol{o}^+)\right] = \sum_{\boldsymbol{o} \text{ is } \boldsymbol{o}^+} \frac{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot \mathbb{1}(\boldsymbol{o} \text{ is } \boldsymbol{o}^+)}{\sum_{\boldsymbol{o} \text{ is } \boldsymbol{o}^+} \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})} = 1. \tag{7}$$

Since $\max_{\pi} \mathbb{E}_{\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)}\left[\mathbb{1}(\boldsymbol{o} \text{ is } \boldsymbol{o}^+)\right] = 1$, we know that $\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) = P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)$ is one of the optimal solutions (not unique), i.e., $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+) \in \arg\max_{\pi} \mathbb{E}_{\pi(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}^+)}\left[\mathbb{1}(\boldsymbol{o} \text{ is } \boldsymbol{o}^+)\right]$. □

B.2 Choice of the Feedback Distribution in Eq. (1)

In Eq. (1), the expectation over $\boldsymbol{c}$ is taken w.r.t. $P_{\text{off}}(\boldsymbol{c}|\boldsymbol{x})$. In principle, any other distribution $p(\boldsymbol{c}|\boldsymbol{x})$ could be used, and the optimal solution $\pi^*_\theta(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c}) = P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x}, \boldsymbol{c})$ would remain unchanged on

$\mathrm{supp}(p(\cdot|\boldsymbol{x}))$. We adopt $P_{\text{off}}(\boldsymbol{c}|\boldsymbol{x})$ for two reasons. **(i)** Its support

$$\mathrm{supp}(P_{\text{off}}(\cdot|\boldsymbol{x})) = \bigcup_{\boldsymbol{o}\in\mathrm{supp}(\pi_{\text{ref}}(\cdot|\boldsymbol{x}))} \mathrm{supp}(p_{\text{env}}(\cdot|\boldsymbol{x},\boldsymbol{o}))$$

covers all feedback that may be encountered when collecting offline data. **(ii)** It serves as a compensating distribution that converts the intractable expectation under the posterior $P_{\text{off}}(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})$ into a tractable expectation under the joint $P_{\text{off}}(\boldsymbol{o},\boldsymbol{c}|\boldsymbol{x}) = \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x}) \cdot p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})$, which is convenient to sample from.

B.3 Connection to Inverse Dynamics Modeling

In traditional RL, objectives for representation learning are often grouped into three classes: **behavior cloning**, **forward dynamics**, and **inverse dynamics**. Behavior cloning is typically used for imitation learning (Arora et al., 2020; Zang et al., 2022), forward dynamics is central to world modeling (Ha & Schmidhuber, 2018; Schwarzer et al., 2021), and inverse dynamics has been explored for both pretraining (Brandfonbrener et al., 2023) and feature extraction for exploration in RL (Efroni et al., 2021).

Interestingly, analogous structures appear in the LLM literature. The objectives of supervised finetuning (SFT), critique finetuning (CFT) (Wang et al., 2025a), and our feedback-conditional policy (FCP) align naturally with behavior cloning, forward dynamics, and inverse dynamics, respectively:

$$\begin{aligned}
\textbf{SFT (behavior cloning):}\quad & \max_{\pi_\theta} \mathbb{E}_{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})}\left[\log \pi_\theta(\boldsymbol{o}|\boldsymbol{x})\right]; \\
\textbf{CFT (forward dynamics):}\quad & \max_{\pi_\theta} \mathbb{E}_{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})}\left[\mathbb{E}_{p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})}\left[\log \pi_\theta(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})\right]\right]; \\
\textbf{Our FCP (inverse dynamics):}\quad & \max_{\pi_\theta} \mathbb{E}_{\pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})}\left[\mathbb{E}_{p_{\text{env}}(\boldsymbol{c}|\boldsymbol{x},\boldsymbol{o})}\left[\log \pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})\right]\right].
\end{aligned} \tag{8}$$

We further illustrate this categorization with graphical models in Figure 3. This unified perspective clarifies the conditional structure underlying each finetuning paradigm and highlights how different forms of supervision drive model learning. In particular, our FCP extends the analogy by treating verbal feedback as a first-class supervision signal, positioning it as the natural *inverse-dynamics* counterpart to existing finetuning objectives.

B.4 Connection to Goal-Conditioned RL

FCP is related to prior goal-conditioned and reward-conditioned policy learning methods (Andrychowicz et al., 2017; Kumar et al., 2019; Zheng et al., 2022), which also learn conditional policies. The key difference is the conditioning signal: prior methods typically condition on goals, achieved goals, or scalar returns, whereas FCP conditions directly on textual feedback $\boldsymbol{c}$. Thus, while FCP shares the high-level template of conditioning policy behavior on a desired signal, it replaces scalar or state-based goals with free-form verbal critiques and relies on the model's language understanding to associate responses with feedback.

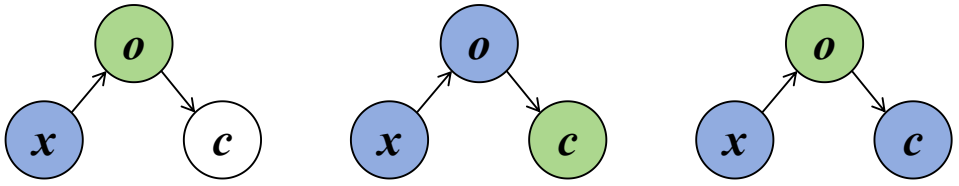
B.5 Why Negative Feedback Can Be Used for Learning

A natural question is why feedback associated with incorrect or low-quality responses can still be useful for training. In FCP, this effect is implicit rather than explicit. Negative feedback is not treated as a scalar penalty or a negative reward. Instead, following Section 2.1, it is simply an observed value of the condition variable $\boldsymbol{c}$ in the feedback-conditional policy $\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c})$. Training on a negative example $(\boldsymbol{x},\boldsymbol{o}^-,\boldsymbol{c}^-)$ therefore teaches the model that the response $\boldsymbol{o}^-$ is associated with critiques of type $\boldsymbol{c}^-$.

This does not by itself impose an explicit RL-style penalty on $\boldsymbol{o}^-$ under a positive condition $\boldsymbol{c}^+$. Rather, the suppression of low-quality responses is implicit in the FCP factorization:

$$\pi_\theta(\boldsymbol{o}|\boldsymbol{x},\boldsymbol{c}^+) \propto \pi_{\text{ref}}(\boldsymbol{o}|\boldsymbol{x})\, p_{\text{env}}(\boldsymbol{c}^+|\boldsymbol{x},\boldsymbol{o}).$$

If a response $\boldsymbol{o}^-$ is unlikely to elicit the desired positive feedback $\boldsymbol{c}^+$, then it receives lower probability mass when the model is conditioned on $\boldsymbol{c}^+$. Thus, negative examples remain useful because they help the model distinguish outputs that are compatible with positive feedback from those that are not, even though FCP does not apply an explicit negative-advantage update.



(a) SFT (*behavior cloning*) (b) CFT (*forward dynamics*) (c) **FCP** (*inverse dynamics*)

Figure 3: **Graphical models for SFT, CFT, and our FCP.** Following Brandfonbrener et al. (2023), we use blue color to indicate inputs to the algorithm and green color to indicate prediction targets.

Table 8: Examples of feedback in *simulated user*-style and *professional reviewer*-style.

Role	Critique Type	Examples
Simulated User	`fully_positive`	That looks right to me, concise and easy to follow. I'm satisfied with the final result.
	`fully_negative`	I have no idea what you were trying to say—the response is nonsense and not helpful at all.
	`neutral`	I'm not completely sure about the logic, but the final answer matches the number I was expecting.
Professional Reviewer	`fully_positive`	Correct and clear; succinct and logically sound, with concise and effective reasoning.
	`fully_negative`	Incorrectly structured and incoherent. The reasoning is absent and the content is unusable.
	`neutral`	Correct final result but unclear and incomplete reasoning; concise yet insufficiently rigorous.

Equivalently, FCP learns a conditional correspondence between responses and feedback, rather than a scalar preference ordering over responses. Since both positive and negative critiques are treated uniformly as values of $\boldsymbol{c}$, learning from $(\boldsymbol{x}, \boldsymbol{o}^-, \boldsymbol{c}^-)$ does not conflict with learning under positive conditions. It instead improves the model's ability to separate different regions of the response space according to the feedback they are likely to receive.

C More Experiment Results

C.1 Additional Training Design Choices

We further study how different design choices affect FCP training, with results summarized in Table 9.

Online update strategy. Compared to the fully online setup, using a larger prompt batch size of 2048 while keeping the mini-batch size fixed at 512 results in four gradient updates per rollout step, and yields better accuracy. This suggests that partial online updates can improve optimization efficiency.

Loss aggregation. In Algorithm 2, cross-entropy on self-sampled responses reduces to policy gradient with unit advantages, which suffers from length bias (Liu et al., 2025a). A debiased scheme averaging at the sequence level and summing at the token level gives a +0.7% gain.

Other variations. We also experimented with several alternative configurations. Reducing the training batch size to 256 improves accuracy by about +2.4%. Training the online stage on a dataset different from that used for offline pretraining slightly underperforms the default baseline, yet remains +5.8% above the offline-only initialization, indicating that offline and online datasets need not be strictly aligned for FCP to be effective.

Table 9: Ablation results on hyperparameter choices, data sources, and feedback settings. Reported numbers are average accuracy on math benchmarks; Δ shows change relative to the default setting.

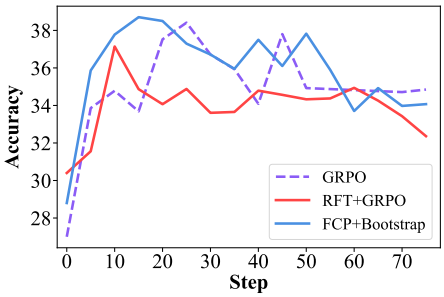
Variant	Changed Setting(s)	Avg Acc	Δ
Default	——	35.3	0.0
w/ partial online	`prompt_bsz`=2048	38.7	+3.4
w/ unbiased loss	`loss_agg_mode`=seq-mean-token-sum	36.0	+0.7
w/ smaller batch size	`train_bsz`=`ppo_mini_bsz`=256	37.7	+2.4
w/ more diverse c^+	use random c^+ per rollout	34.1	-1.2
w/ different dataset	use MATH-Train split	34.3	-1.0

Table 10: **Backbone comparison.** Mean math accuracy (%). Qwen columns average five tasks; Llama averages MATH500, Minerva and Olympiad only. Compare methods within each column.

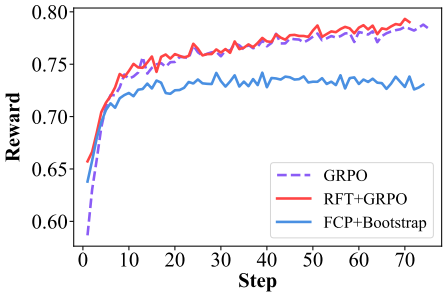
Method	Qwen2.5-7B	Qwen3-4B	Llama-3.1-8B
Base	27.0	28.3	6.2
Base + GRPO	38.4	37.8	10.3
RFT	30.4	32.2	8.5
RFT + GRPO	37.1	34.7	10.8
FCP	28.5	32.6	7.6
FCP + Bootstrap	38.7	36.2	10.7

C.2 Backbone Model Sensitivity

To verify that FCP generalizes across different model families and scales, Table 10 extends our math evaluation to `Qwen3-4B-Base` (Yang et al., 2025a) and `Llama-3.1-8B` (Grattafiori et al., 2024). Across both architectures, FCP + Bootstrap yields substantial improvements over the base models and consistently performs on par with scalar-reward GRPO baselines. Specifically, on Qwen3-4B, FCP + Bootstrap achieves 36.2% average accuracy, closely matching Base + GRPO (37.8%) and outperforming RFT + GRPO (34.7%). On Llama-3.1-8B, FCP + Bootstrap reaches 10.7% (evaluated on MATH500, Minerva, and OlympiadBench, as all methods show floor effects on AIME), directly on par with Base + GRPO (10.3%) and RFT + GRPO (10.8%). These results confirm that FCP's ability to learn directly from verbal feedback is not confined to a single model family, but generalizes reliably across diverse architectures.



(a) Average accuracy over five math benchmarks, measured w.r.t. training steps.



(b) Scalar rewards assigned by `GPT-5-nano` to model rollouts, measured w.r.t. training steps.

Figure 4: **Training dynamics of FCP and baselines.** (a) FCP+Bootstrap matches GRPO and RFT+GRPO accuracy within 30 steps. (b) FCP's scalar reward scores lag behind baselines, consistent with the fact that FCP does not directly optimize against reward models.

Table 11: **IFEval (out-of-distribution) results.**

Offline Algo. + Online Algo.	Env. Model	Scalar Reward	OOD IFEval
Base	-	-	36.8
+ GRPO	LLM	✓	38.5
+ Critique-GRPO	LLM+RBV	✓	**39.0**
RFT	RBV	✓	37.5
+ GRPO	LLM	✓	38.8
+ Critique-GRPO	LLM+RBV	✓	38.6
FCP	LLM	✗	38.6
+ Bootstrap	LLM	✗	**39.0**

Table 12: Evaluation under In-Distribution (ID) and Out-of-Distribution (OOD) test-time feedback. ID uses reviewer-style feedback matching training, while OOD use mismatched or absent feedback.

Method	Feedback Style	ID/OOD	AIME24	AIME25	MATH500	Minerva	Olympiad	Avg.
Base	-	-	7.5 ±1.7	6.7 ±0.0	63.8 ±0.4	28.3 ±0.8	28.6 ±0.4	27.0 ±0.5
FCP + Bootstrap	Reviewer	ID	25.0 ±3.3	7.5 ±1.7	76.5 ±0.7	45.8 ±0.7	38.8 ±0.6	38.7 ±0.7
FCP + Bootstrap	User	OOD	10.8 ±4.2	7.5 ±1.7	75.1 ±1.0	35.2 ±0.2	38.0 ±0.4	33.3 ±1.1
FCP + Bootstrap	Human	OOD	13.3 ±2.7	7.5 ±3.2	75.2 ±0.4	34.2 ±0.7	38.1 ±0.4	33.6 ±0.7
FCP + Bootstrap	"Correct"	OOD	16.7 ±4.7	7.5 ±1.7	75.3 ±0.5	35.2 ±0.2	37.2 ±0.9	34.4 ±1.0
FCP + Bootstrap	No Feedback	OOD	15.8 ±1.7	10.0 ±4.7	74.4 ±0.7	35.6 ±0.4	37.6 ±0.9	34.7 ±1.2

C.3 Learning Dynamics of FCP

FCP achieves strong accuracy without over-optimizing scalar rewards. As seen in Figure 4a, both FCP and GRPO reach peak accuracy within 30 online steps, with scalar scores from `GPT-5-nano` rising sharply at the start. Yet Figure 4b shows FCP's scores lagging behind GRPO's later, since it does not directly optimize against the scalar reward model. Crucially, FCP sustains high accuracy despite lower scores, suggesting that verbal feedback can provide an effective training signal beyond direct scalar reward maximization.

C.4 Instruction Following Generalization on IFEval

Although FCP is trained on math tasks, we also evaluate its instruction-following generalization on the out-of-distribution IFEval benchmark. As shown in Table 11, FCP and FCP+Bootstrap reach 38.6% and 39.0%, respectively, comparable to GRPO (38.5%) and RFT+GRPO (38.8%), and matching the best scalar-based baseline (39.0% from Critique-GRPO). These results suggest that FCP's training does not impair, and may even slightly improve, instruction-following performance on OOD prompts.

C.5 Effect of OOD Feedback at Test Time

At test time, FCP can condition on user-provided feedback. A natural question is how the model behaves when this feedback is out of distribution or unavailable. To probe this, we evaluate the FCP checkpoint under several OOD feedback settings: a simulated *user* setting using the same feedback style as in Section 4.1, a *human* setting using human-written feedback from WildChat and LMSYS-Chat (Zhao et al., 2024; Zheng et al., 2023), a single-word *"Correct"* setting, and a *no-feedback* setting where the feedback slot is left empty.

As shown in Table 12, FCP remains substantially stronger than the base model across all OOD settings. In terms of average accuracy, it achieves 33.3% with simulated *user* feedback, 33.6% with *human* feedback, 34.4% with single-word *"Correct"* feedback, and 34.7% with *no-feedback*, compared to 27.0% for the base model. These results indicate that FCP retains much of its gains even under substantial shifts in the test-time feedback distribution.

We hypothesize that this robustness arises in part because the system and user prompts already encode implicit expectations of helpful and correct behavior (e.g., default instructions such as "You are a

helpful assistant"), so the model can continue to leverage the learned feedback-conditional policy even when explicit or in-distribution feedback is absent.

Table 13: **Hyperparameters for FCP training** used in the offline and bootstrapping (online) stages.

Hyperparameter	Offline	Online
learning rate	5e-6	1e-6
lr scheduler	cosine	constant
weight decay	0	0.01
warmup ratio	0.1	0
train batch size	512	2048
ppo mini-batch size	—	512
temperature	—	1.0
top_p	—	1.0
rollout_n	—	4
epoch	1	
max response length	4096	
loss type	cross-entropy loss	
loss aggregation mode	token-mean	
feedback environment	`GPT-5-nano`	
feedback style	*professional reviewer*	

D Detailed Experimental Setup

All implementations are based on `llama-factory` (Zheng et al., 2024) and `verl` (Sheng et al., 2025). Hyperparameter settings for both offline and online stages of FCP are listed in Table 13, and the corresponding training costs are reported in Table 14. The full research project involved additional compute for preliminary exploratory experiments and failed trials, which were not systematically logged and are therefore excluded from Table 14.

For the two special tokens `<EF>` and `</EF>`, embeddings are initialized by sampling from a multivariate normal distribution with mean and covariance computed over existing token embeddings. For online conditioning, desired positive feedback conditions are sampled from the training pool; random examples of these conditions are shown in Table 15. For general reasoning bootstrapping, we adopt a fully online setup with batch size of 256, differing from the math setting to illustrate that FCP remains effective under both training strategies.

Figure 5 shows the unified prompt template used to elicit feedback from `GPT-5-nano`. The template produces three outputs in one response: a less detailed *simulated user*-style feedback, a high-quality *professional reviewer*-style feedback, and a scalar score summarizing overall quality.

Table 14: Comparison of training cost across different methods. All experiments are conducted on 8 NVIDIA A100 40GB GPUs.

Method	**Training Time (Hours)**	**FLOPs**
Base	-	-
GRPO	7.8	69.8
RFT	3.7	32.9
RFT+GRPO	8.0	71.5
FCP	3.6	32.6
FCP+Bootstrap	7.1	63.5

Table 15: Random samples of high-level positive feedback conditions vc^+ from the training pool.

Examples
Correct and clear. Thorough, logical, and concise.
Clear and correct. Thorough yet concise, with straightforward reasoning and a correct result.
Correct, clear, and straightforwardly reasoned. The reasoning is sound and the result is accurate.
Correct and clear. The reasoning is sound and complete, and the presentation is concise and readable.
Correct and coherent. Clear and methodical. Consistent and concise.
Clear and correct. Logical and complete, with accessible reasoning and concise steps.
Correct, clear, and complete. The reasoning is straightforward and well-structured, producing an interpretable result.
accurate and logically sound; clear and complete; concise but readable.

You are acting as a real-world human user of an LLM.

Inputs:
Question:
\"\"\"
{question}
\"\"\"

Model Answer:
\"\"\"
{model_answer}
\"\"\"

Reference Final Answer (used only for correctness check):
\"\"\"
{reference_answer}
\"\"\"

Your tasks:

1) Simulate "user feedback" from a normal, real-world user reacting to the Model Answer only.
- Length: 1-3 sentences, colloquial tone, first person.
- Content: purely subjective sentiment (e.g., helpfulness, confidence, confusion, satisfaction).
- STRICT: Do NOT mention or allude to any symbols, formulas, variable names, or specialized concepts from the Question or the Model Answer. Do NOT quote text from the inputs.

For example:
"I think you are right, but your solution is really long and complicated."
"You are a genius! You have all my respect."
"I am confused. There seems to be a mistake in your solution."
"What are you talking about? You are not answering my question."
etc.

2) Simulate a professional reviewer evaluating the Model Answer along several dimensions, including but not limited to:
• correctness — Compare the Model Answer's final result ONLY against the Reference Final Answer (if provided). Judge whether the end result matches; do not use the reference for any other purpose.
• logical_rigor — Assess the soundness and gaplessness of reasoning within the Model Answer itself. Do NOT use the Reference Final Answer here.
• completeness — Judge coverage of required parts and edge cases based on the Question and the Model Answer only. Do NOT use the Reference Final Answer here.
• clarity — Evaluate organization, readability, and ease of following in the Model Answer. Do NOT use the Reference Final Answer here.

Then provide a high-level summary (1-3 sentences) with overall judgment and broad observations.
- STRICT for the high-level summary: Only use adjectives and adverbs to describe the Model Answer and reasoning process. DO NOT mention where it goes wrong and where it can do better.

For example:
"Your final answer is correct, but the solution is too long and complicated. There are also several logical errors in your solution."
"The answer is partially correct. The reasoning is sound but not complete. Also, you are being too verbose."
"The answer is totally wrong. It lacks soundness and is not complete. However, the solution is concise and clear."

Hard constraints:
- Keep all content in English.
- Do not mention anything like "reference" or "python snippet".

Output format:
User-style feedback: <your 1-3 sentence feedback>
Analysis along several dimensions: <your 1-3 sentence analysis>
High-level summary: <your 1-3 sentence summary>
Score (0-10): <one overall integer score>

Figure 5: Prompt template used to elicit feedback from `GPT-5-nano`, including *simulated user*-style feedback, *professional reviewer*-style feedback, and a scalar score.

You are a professional creative-writing reviewer evaluating a story's quality.

Your tasks:

First, reason carefully and extensively about the Story before producing the final JSON. Analyze its narrative construction, strengths, weaknesses, and concrete evidence for each dimension step by step. Resolve any conflicts between positive and negative evidence before assigning scores.

After completing the reasoning, perform the following tasks:

1) Evaluate the Story as a professional creative-writing reviewer along the following dimensions.

For every dimension:
- Evaluate each dimension independently from the other dimensions.
- Base the evaluation only on the Story.
- Do not invent details that are not present.
- Provide one integer score from 0 to 10.
- A score of 5 means average but functional.
- Scores of 9 or 10 should be rare and reserved for exceptional writing.
- Do not reward the Story merely for being long.
- Do not penalize the Story merely for being short if it is complete and effective.

Canonical positive-feedback rule:
- If a dimension is good overall, assign a score from 6 to 10 and set its feedback to exactly "Perfect."
- When using "Perfect.", output only that exact word and period. Do not add praise, explanation, qualifications, or revision advice.
- For a score from 0 to 5, provide 1-3 sentences of specific, actionable corrective feedback.
- Corrective feedback should identify the most important weakness in that dimension and explain how the story should be improved.
- Never use "Perfect." for a score from 0 to 5.
- Never provide corrective feedback for a score from 6 to 10.

Dimensions:

A. narrative_execution
Evaluate whether the Story works effectively as a narrative. Consider plot development, causal progression, coherence, conflict, pacing, tension, use of narrative space, number of characters, stability of point of view, completeness, and whether the ending follows from the earlier development and provides an effective payoff.

B. characterization_and_emotional_impact
Evaluate whether the characters have understandable motivations, distinct personalities, meaningful reactions, and enough emotional depth for the reader to care about what happens.

C. originality_and_creativity
Evaluate whether the Story develops its premise in a distinctive and imaginative way rather than relying mainly on generic plots, stock characters, clichés, or predictable moral lessons.

D. prose_and_style
Evaluate diction, sentence flow, imagery, dialogue, paragraph structure, readability, grammar, and overall stylistic control.

2) Identify validity problems.

Set is_valid_story to false if the response:
- is not primarily a creative story;
- is severely incomplete or abruptly truncated;
- contains major unrelated continuation;
- is dominated by code, instructions, analysis, or chat-template text;
- contains severe repetition that prevents normal reading.

Possible flags:
- "not_a_story"
- "incomplete"
- "unrelated_continuation"
- "code_or_instruction_contamination"
- "chat_template_contamination"
- "severe_repetition"
- "confusing_point_of_view"
- "too_many_characters"
- "generic_or_template_like"

Use an empty list if no flag applies.

Hard constraints:
- Keep all output content in English.
- Use the model's reasoning section for the detailed chain-of-thought analysis.
- After the reasoning section, return exactly one valid JSON object as the final answer.
- Do not wrap the final JSON object in Markdown code fences.
- Do not include any content after the final JSON object.
- Every score must be an integer from 0 to 10.
- Each dimension must contain exactly one "feedback" string and one "score" integer.

Output format:

```
{{
  "validity": {{
    "is_valid_story": true,
    "flags": []
  }},
  "dimensions": {{
    "narrative_execution": {{
      "feedback": "<Perfect. OR 1-3 sentence actionable corrective
feedback>",
      "score": <integer from 0 to 10>
    }},
    "characterization_and_emotional_impact": {{
      "feedback": "<Perfect. OR 1-3 sentence actionable corrective
feedback>",
      "score": <integer from 0 to 10>
    }},
    "originality_and_creativity": {{
      "feedback": "<Perfect. OR 1-3 sentence actionable corrective
feedback>",
      "score": <integer from 0 to 10>
    }},
    "prose_and_style": {{
      "feedback": "<Perfect. OR 1-3 sentence actionable corrective
feedback>",
      "score": <integer from 0 to 10>
    }}
  }}
}}
```

Story:
<story>
{story}
</story>

Figure 6: Prompt template used to elicit multi-dimensional creative-writing feedback from `Qwen3.5-122B-A10B` on LitBench, covering narrative execution, characterization, originality, prose style, validity flags, and dimensional scores.