BRINGING VALUE MODELS BACK: GENERATIVE CRITICS FOR LLM REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Credit assignment is a central challenge in reinforcement learning (RL). Classical actor-critic methods address this challenge through fine-grained advantage estimation based on a learned value function. However, learned value models are often avoided in modern large language model (LLM) RL because conventional discriminative critics are difficult to train reliably. We revisit value modeling and argue that this difficulty is partly due to limited expressiveness. In particular, representation complexity theory suggests that value functions can be hard to approximate under the one-shot prediction paradigm used by existing value models, and our scaling experiments show that such critics do not improve reliably with scale. Motivated by this observation, we propose Generative Actor-Critic (GenAC), which replaces one-shot scalar value prediction with a generative critic that performs chain-of-thought reasoning before producing a value estimate. We further introduce In-Context Conditioning, which helps the critic remain calibrated to the current actor throughout training. GenAC improves value approximation, ranking reliability, and out-of-distribution generalization, and these gains translate into stronger downstream RL performance than both value-based and value-free baselines. Overall, our results suggest that stronger value modeling is a promising direction for improving credit assignment in LLM reinforcement learning.

1 INTRODUCTION

Reinforcement learning (RL) has emerged as a cornerstone of large language model (LLM) post-training, transforming next-token predictors into capable assistants that solve complex problems (Jaech et al., 2024; Guo et al., 2025a; Team et al., 2025). Initially popularized through human value alignment (Ouyang et al., 2022; Rafailov et al., 2023), RL is now widely deployed to enhance mathematical reasoning (Shao et al., 2024; Guo et al., 2025a), code generation (Le et al., 2022), and agentic systems (Team et al., 2025). As these applications demand longer-horizon reasoning and more complex interactions, the ability of the underlying RL algorithms to provide effective training signals becomes a critical bottleneck.

A central challenge in RL is *credit assignment*: determining which decisions within a long sequence contributed to an eventual outcome (Sutton & Barto, 1998). This challenge is particularly pronounced in LLM post-training, where supervision is typically limited to a single scalar reward at the end of an entire sequence. Actor-critic methods (Konda & Tsitsiklis, 1999) address credit assignment by learning a value function that converts sparse outcome rewards into dense, token-level advantages, providing more informative training signals for the policy. Among these methods, Proximal Policy Optimization (PPO) (Schulman et al., 2017) provides a practical and influential instantiation of this idea, combining stable clipped-surrogate optimization with fine-grained advantage estimation via Generalized Advantage Estimation (GAE) (Schulman et al., 2015). This value-based recipe has been proven effective in various applications including LLM alignment (Ouyang et al., 2022).

Yet in recent practice, PPO-style methods have fallen out of favor. Training an accurate value model at LLM scale is both difficult and computationally expensive, pushing practitioners toward simpler value-free alternatives. Methods such as Group Relative Policy Optimization (GRPO) (Shao et al., 2024) and REINFORCE Leave-One-Out (RLOO) (Ahmadian et al., 2024) bypass the critic entirely by estimating uniform advantages from multiple sampled trajectories, and have gained wide popularity as a practical workaround that achieves competitive performance.

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.

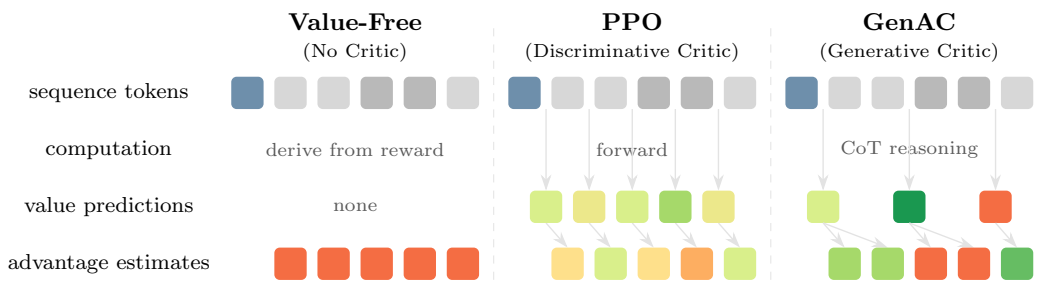


Figure 1: Advantage estimates from value-free methods, PPO, and GenAC on a trajectory with a correct prefix followed by an error and an incorrect final answer.

However, this simplicity comes at the cost of precise credit assignment. Without a value function, these methods assign uniform credit across all tokens in a trajectory, effectively reducing sequential generation to a contextual bandit problem (Langford & Zhang, 2007). This over-simplification could cap the performance ceiling of LLM post-training, and the issue grows as the field increasingly targets longer context applications such as complex reasoning, multi-turn conversations and agentic systems. This motivates us to revisit value modeling in LLM RL. Rather than viewing the recent shift toward value-free methods as evidence that critics are unnecessary, we ask: *can value models be made accurate and robust enough to restore fine-grained credit assignment in practice?*

Our primary contributions are as follows.

1. We analyze the limitations of standard discriminative critics in LLM RL from both theoretical and empirical perspectives. Drawing on representation complexity theory, we provide evidence that their difficulty arises in part from limited expressiveness under the one-shot scalar prediction paradigm, rather than from optimization challenges alone. Controlled scaling experiments confirm that their approximation error does not improve reliably with model size and is highly sensitive to random seeds.
2. We propose Generative Actor-Critic (GenAC), which replaces one-shot scalar value prediction with a generative critic that performs explicit chain-of-thought reasoning before producing value estimates, directly addressing the expressiveness limitation above. We further introduce In-Context Conditioning, a mechanism that keeps the critic calibrated to the current actor throughout training.
3. On mathematical reasoning benchmarks, GenAC achieves superior sample efficiency and continues to improve in training regimes where baselines plateau. Supporting analyses show that the generative critic ranks candidate actions more reliably, generalizes better out-of-distribution, and produces interpretable credit assignments that are consistent with reasoning quality.

Related Works. Existing methods differ in how they derive training signals from outcome rewards. Value-free methods estimate uniform advantages from trajectory rewards (Shao et al., 2024; Ahmadian et al., 2024) or recover fine-grained feedback through costly partial rollouts (Kazemnejad et al., 2025; Guo et al., 2025c). VC-PPO (Yuan et al., 2025b) improves value modeling through changes to critic training rather than to the underlying discriminative architecture. Another line of work improves reward modeling by scoring intermediate reasoning steps with process reward models (PRMs) (Lightman et al., 2023; Wang et al., 2024; Yuan et al., 2025a), or using chain-of-thought reasoning before producing assessments with generative reward models (Zhang et al., 2024; Mahan et al., 2024; Liu et al., 2025b; Chen et al., 2025b) and generative PRMs (Zhao et al., 2026; Khalifa et al., 2025). Our work instead implements credit assignment through value baselines that leave the RL objective unchanged, avoiding reward hacking incentives while allowing greater tolerance for prediction errors. Moreover, value models must continually adapt to track the value function of an evolving policy. We include a full discussion in Appendix A.

2 Preliminaries

Language Generation as an MDP. Language generation can be modeled as a Markov Decision Process $(\mathcal{S}, \mathcal{A}, P, R, \rho, T)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is the transition

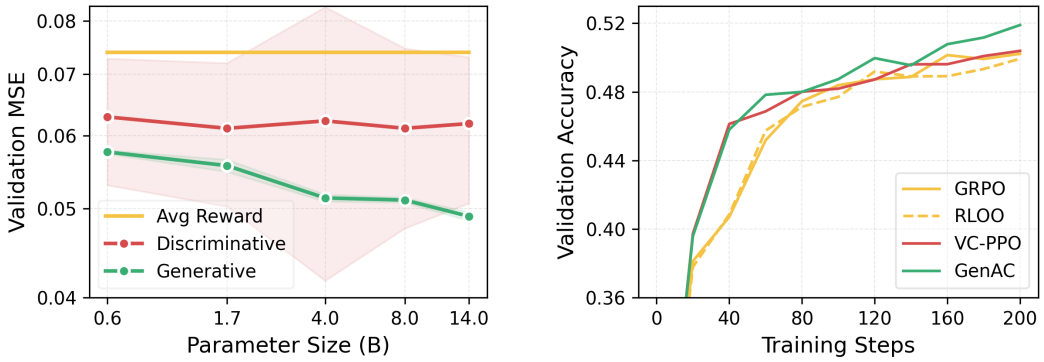


Figure 2: **(a)** Validation MSE when fitting the same fixed value function across critic sizes; mean and standard deviation over 10 seeds. `Avg Reward` predicts the mean reward of eight trajectories for every state. **(b)** RL validation accuracy during training, averaged over 16 generations per problem and six mathematical reasoning benchmarks.

dynamics, R is the reward function, ρ is the distribution of initial states, and T is the maximum number of decision steps. At step t, the state $s_t = x \circ y_{<t} \in \mathcal{S}$ is the concatenation of the prompt x and the response generated so far $y_{<t} = y_1 \circ y_2 \circ \cdots \circ y_{t-1}$; the action $a_t \in \mathcal{A}$ is the new generation at granularity of token, segment or response; the transition dynamics P is deterministic, appending the new generation to the state to obtain the next state $s_{t+1} = P(s_t, a_t) = s_t \circ a_t$; the reward function R assigns 0 to all steps except the last, where it provides a binary reward depending on the correctness of the generation. A parameterized policy $\pi_\theta(a|s)$ determines the probability of taking action a at state s. Its value function $V(s)$ is defined by the expected cumulative reward if starting at state s then following π. Its Q function $Q(s, a)$ is defined by the expected cumulative reward if starting at state s, taking action a, then following π. The advantage function $A(s, a)$, defined by $Q(s, a) - V(s)$, depicts the expected improvement in cumulative reward from taking action a over the average action at state s, then following π.

Proximal Policy Optimization. PPO and its variants are the most widely used RL algorithms in LLMs. It is efficient through minibatch update and stable by optimizing a clipped surrogate objective

$$\mathcal{L}_{\text{PPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_{\theta_{\text{old}}}} \left[-\sum_{t=1}^{|y|} \min\left(\rho_t(\theta)\hat{A}_t, \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right]$$

where $\rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$, $\hat{A}_t$ is the advantage estimation at step t, and ϵ is a small hyperparameter. Variants of PPO differ in how $\hat{A}_t$ is computed. PPO estimates token-level advantages via GAE (Schulman et al., 2015):

$$\hat{A}_t = \sum_{k=0}^{T-t} (\gamma\lambda)^k (r_{t+k} + \gamma v_\phi(s_{t+k+1}) - v_\phi(s_{t+k}))$$

where v_ϕ is a learned critic model, r_t is the observed reward at step t, γ is the discount factor, and λ is a hyperparameter. The critic model is trained to minimize MSE towards TD(λ) targets on the same data distribution. GRPO and RLOO estimate trajectory-level advantages utilizing the rewards of all responses for the same prompt. GRPO normalizes the rewards within a group $\hat{A}_{i,t} = \frac{r_i - \text{mean}_j r_j}{\text{std}_j r_j}$, while RLOO simply introduces a sample-based baseline $\hat{A}_{i,t} = r_i - \text{mean}_{j,j\neq i} r_j$.

3 Limitations of Discriminative Critics

We investigate the limitations of standard discriminative critics in LLM RL from both theoretical and empirical perspectives. Our analysis suggests that the difficulty of value learning is not purely an optimization problem, but is also rooted in the limited expressiveness of the one-shot scalar prediction paradigm. These findings motivate the more expressive generative critic developed in the next section.

A Representation-Complexity Perspective. We study the limitations of standard discriminative critics through the lens of *representation complexity*. The key issue is not only that value prediction can be noisy or difficult to optimize, but also that the value function itself may be intrinsically difficult to represent with a one-shot scalar predictor. A standard discriminative critic maps a prompt together with a partial response directly to a scalar value in a single forward pass of fixed depth. This raises a natural question: even under stable optimization, is such a predictor expressive enough for value functions arising in sparse-reward language-generation MDPs? Recent theory suggests that the answer can be no. In particular, Feng & Zhong (2024) construct a family of MDPs parameterized by an underlying language $L \in \mathsf{P}$, and show that the associated value-function computation can already be P-complete. This is not a single pathological example, but a fairly general construction template indexed by the underlying language. Meanwhile, log-precision Transformers with constant depth and polynomial hidden dimension are confined to the complexity class TC^0 (Merrill & Sabharwal, 2023), which is widely believed to be strictly weaker than P. The implication is that value functions can be substantially harder to represent than a standard scalar critic can express. From this perspective, the difficulty of value learning may stem not only from optimization, but also from a mismatch between the target value function and the expressive capacity of one-shot scalar prediction. Notably, since log-precision Transformers of any practical scale remain within TC^0, this perspective indicates that increasing model size alone, without adding sequential computation, should yield limited improvement, a prediction we test empirically below. With chain-of-thought, however, log-precision Transformers can characterize exactly P (Feng et al., 2023; Li et al., 2024), which motivates the use of chain-of-thought generation.[1]

Empirical Approximation Performance. To probe this representation-complexity perspective empirically, we isolate value estimation from the rest of RL and fit a fixed value function, namely that of the initial policy `Qwen3-8B-Base` (Yang et al., 2025). Analogous to the value pretraining stage in Yuan et al. (2025b), we freeze the actor and train the critic on a held-out split of the DeepScaleR (Luo et al., 2025) prompt dataset. All critics are initialized directly from the Qwen3 base series. Following Yuan et al. (2025b), we formulate the MDP at the token level and fit Monte Carlo returns. We evaluate on a curated benchmark of state-value pairs, where each state consists of a validation prompt together with a partial response generated by the initial policy and truncated at semantic boundaries. Following Wang et al. (2024), we estimate the ground-truth value of each state by averaging the final rewards over multiple continuations, and report the mean squared error between these labels and the critic predictions. To study scaling behavior, we train critics across model sizes ranging from 0.6B to 14B. For each size, we tune the learning rate and the amount of training data to obtain stable convergence. We repeat each setup 10 times with different random seeds and report the mean and standard deviation of the final validation MSE in Figure 2(a).

- **Not scalable.** Despite substantial increases in pretrained model capacity and training compute, critics across all evaluated sizes achieve remarkably similar approximation errors. This suggests that larger discriminative critics still fail to realize materially better value representations.
- **Not robust.** Final performance varies sharply across random seeds for every model size. Since LLMs are typically much less sensitive to such randomness, this variation points to pronounced instability in discriminative value approximation.

Our theoretical analysis motivates additional sequential computation, while our experiments expose limitations in the scaling and robustness of discriminative critics. Together, these findings suggest that value estimation may benefit from a more expressive architecture, motivating the intermediate-reasoning critic developed in the next section.

4 Generative Actor-Critic

The representation-complexity perspective developed in Section 3 clarifies why value modeling may require a more expressive critic parameterization. The appeal of a generative critic is not simply that it produces text before a scalar value, but that it preserves autoregressive generation and thereby enables intermediate reasoning. Recent work further suggests that such intermediate generation can strictly expand transformer expressive power relative to one-shot prediction (Feng et al., 2023; Li

[1] For readers less familiar with the complexity notation, Appendix B provides a brief introduction.

Prompt

A 8B language model is solving a math problem. Your job is to evaluate its probability of success, given the partial solution it has already generated. For your reference, the average success rate of the model on the dataset is 0.29.
<problem>In the center of a square...</problem>
<partial_solution>The problem describes...</partial_solution>
Instructions:
1. Evaluate the difficulty of the problem.
2. Recover the capabilities of the model based on the partial solution.
3. Skim through the partial solution and detect any progress, error, or confusion.
4. Analyze the probability of success if the model finish the solution.
5. Output the final answer as an integer between 0 and 10 inclusive (10 most likely).

Figure 3: Our proposed prompt template. ICC hints are colored in red.

et al., 2024). If value modeling is fundamentally constrained by representation complexity, then this additional reasoning channel provides exactly the kind of computation that a critic should exploit.

4.1 Generative Critic

We now describe two core components of our generative critic: an autoregressive architecture that enables chain-of-thought reasoning before value estimation, and an in-context mechanism for explicit policy conditioning. Together, they form the foundation for the training pipeline in Section 4.2.

Architecture. Standard discriminative critics replace the pretrained language modeling head with a linear projection layer that maps the final hidden states directly to a scalar value prediction. In contrast, our proposed generative critic retains the original language modeling head, fully leveraging pretrained knowledge and allowing the critic to perform explicit chain-of-thought reasoning before arriving at a final value estimate. Instead of prompting the model to verbalize an arbitrary real-valued scalar, we ask the critic to produce an integer score from 0 to 10 representing the likelihood of success, which we parse and normalize into a value prediction between 0 and 1. This scoring format is inherently more natural for a language model to reason over and predict. We formulate the MDP at the segment level: each response is split into contiguous segments based on heuristic rules, and each segment is treated as a single action. We illustrate the advantage estimation computation of value-free methods, discriminative critics, and generative critics in Figure 1.

In-Context Conditioning (ICC). A fundamental distinction between reward and value functions is that the latter are conditioned on policies, suggesting that critics must be *aware* of the specific capabilities of the actor. While discriminative critics encode all relevant information in their weights, generative critics, as LLMs with in-context learning capabilities, can additionally exploit explicit policy descriptors provided in context. To this end, we design a prompt template that a) explicitly prompts the critic to infer the actor's capabilities from the partial response, and b) provides additional information about the actor, including its model size and a smoothed running average of achieved rewards, as shown in Figure 3. This grounds the critic's reasoning, making a general function approximator more tightly policy-conditioned without requiring it to encode all related information in weights alone. This is particularly important because, as we show in Section 5.3.1, even a strong LLM like GPT-5 fails to serve as an accurate value model without sufficient knowledge of the actor. Our ablation confirms that ICC yields consistent improvements across all training stages.

4.2 Training Pipeline

Prior to the main RL loop, the generative critic undergoes a two-stage pretraining process: (1) *Supervised Finetuning* (SFT), which instills foundational reasoning patterns using synthesized reasoning traces, and (2) *RL Pretraining*, which grounds the critic's reasoning in empirical, ground-truth returns. Following these stages, the critic is fully initialized and ready for joint training with the actor.

SFT. To instill the reasoning capabilities into the generative critic, we construct a dataset of high-quality reasoning traces for cold-start SFT. Using a held-out split of the prompt dataset, we sample partial responses generated by the initial actor, apply the critic prompt template in Figure 3, and query GPT-5 (Singh et al., 2025) to generate reasoning traces. We collect these traces and perform standard SFT on the base model. Notably, despite being a strong language model, GPT-5 is a weak value approximator without sufficient knowledge of the evaluated policy, as Table 2 shows. Thus the purpose of this stage is solely to equip the critic with the reasoning patterns needed for value prediction, while keeping its chain-of-thought within a controlled token budget, rather than distilling value modeling skills from the oracle, making this stage beneficial but not necessary.

RL Pretraining. While SFT successfully instills reasoning capabilities, the resulting critic struggles to produce accurate value estimates due to two reasons. First, SFT models tend to overfit to stylistic patterns in the data and generalize poorly to unseen states (Tajwar et al., 2024; Chu et al., 2025). Second, the critic's performance is inherently upper-bounded by the oracle, which cannot perfectly simulate the initial policy's true performance due to insufficient in-context information (details in Section 5.3.1). To bridge this gap, we freeze the actor and train the critic via RL, directly grounding its reasoning in empirical returns. Specifically, we train the critic using REINFORCE (Williams, 1992) to optimize a rule-based reward. For each state s, the critic generates a reasoning trace z yielding a parsed value prediction $\hat{v}$, which is evaluated against the observed reward r: $R_v(s, z) = 1 - (r - \hat{v})^2$. If parsing $\hat{v}$ from z fails, we set the reward to 0. This reward is motivated by the MSE minimization objective used to train discriminative critics in standard PPO, while additionally enforcing valid output format. Following Yuan et al. (2025b), we set GAE $\lambda = 1$ and remove the KL penalty. Training stops when validation MSE saturates.

RL Joint Training. Once pretrained, the generative critic is integrated into the downstream RL loop. The procedure is identical to RL pretraining, except that the actor is now updated via PPO using advantages computed from the critic's predictions. By alternating between actor and critic updates, the critic continuously adapts its reasoning to track the shifting value function of the evolving actor, ensuring credit assignment remains accurate as the training proceeds. A detailed pseudocode is provided in Appendix C.

Cost Analysis. We conduct a detailed cost analysis in Appendix F. Compared to standard PPO, GenAC incurs approximately $2.1\times$ the computational cost per iteration. However, as we show in Section 5.2, the overhead is offset by improved sample efficiency and continued performance gains when a discriminative critic saturates. Additionally, this cost is an implementation choice, as we choose to explore what is achievable with the strongest value model we can build. We discuss how to reduce cost within our framework, and leave further cost-optimal tuning to future work.

5 Experiments

5.1 Approximation Performance

We repeat the controlled approximation experiment from Section 3, now training generative critics under the same protocol. As shown in Figure 2(a), generative critics achieve substantially lower approximation error than their discriminative counterparts at every model size. Moreover, their error decreases consistently as model size increases and exhibits markedly lower variance across random seeds, addressing both failure modes identified in Section 3.

5.2 RL Experiment

We now present our primary evaluation, conducting extensive RL training experiments on mathematical reasoning to assess the effectiveness of our method.

Setup. We adopt a zero RL setup (Guo et al., 2025a), initializing the policy with `Qwen3-8B-Base` (Yang et al., 2025) and the critic with our pretrained generative critic as detailed in Section 4. Models are trained on the DeepScaleR dataset (Luo et al., 2025) and evaluated across six standard mathematical reasoning benchmarks: AIME 2024, AMC 2023, MATH 500 (Hendrycks et al., 2021), AIME 2025, OlympiadBench (He et al., 2024), and Minerva Math (Lewkowycz et al., 2022). For all evaluations, we report the average accuracy over 16 sampled generations per problem.

Table 1: Final performance in `avg@16` accuracy (%) on validation benchmarks.

Algorithm	AIME24	AMC23	MATH500	AIME25	Olympiad	Minerva	Average
GRPO	26.46	69.69	87.20	**22.92**	52.92	42.12	50.22
RLOO	24.79	68.75	**87.64**	21.25	53.94	43.22	49.93
VC-PPO	25.83	70.78	87.54	21.67	53.50	43.02	50.39
GenAC	**30.00**	**74.53**	87.48	21.88	**54.16**	**43.34**	**51.90**

Baselines. We compare GenAC against three RL baselines: GRPO, RLOO, and VC-PPO. As PPO variants, all algorithms share the same underlying policy optimization framework and differ solely in their advantage estimation mechanisms. GRPO and RLOO serve as value-free baselines that estimate advantages using sample scores, while VC-PPO leverages a pretrained discriminative critic.

Performance of GenAC. From Figure 2(b) and Table 1, our proposed GenAC achieves the most favorable final performance and training dynamics. Notably, it is *sample-efficient*, requiring fewer training steps to outperform baselines. Furthermore, its *performance margin over all baselines continues to grow* as training progresses into the latter half. We attribute this success to the generative critic that provides fine-grained feedback compared to bandit-style baselines, and enables more accurate credit assignment than the standard discriminative critic in VC-PPO.

The Role of Value Modeling. Our comparison of advantage estimators reveals a clear hierarchy in policy performance: GenAC outperforms VC-PPO, which in turn outperforms the bandit-style baselines. Benefiting from fine-grained feedback, both GenAC and VC-PPO are more sample-efficient than GRPO or RLOO during the early stages of training. However, constrained by the limitations of its discriminative critic, VC-PPO's advantage diminishes over time, ultimately resulting in a policy only marginally above the value-free baselines. In contrast, leveraging a more expressive generative critic, GenAC sustains this momentum and achieves an increasing performance margin. This is also supported by our multi-seed replications in Appendix E: GRPO and VC-PPO perform similarly across multiple runs, while GenAC is clearly distinguished from all. These observations suggest that improvements in value modeling *can* translate to superior policy performance in LLM RL, demonstrating that explicit, accurate value modeling remains a critical driver for effective and efficient policy optimization.

5.3 Analysis

Having established the RL training performance, we now take a closer look at the generative critic itself, probing its design choices and characteristic properties through a series of targeted analyses.

5.3.1 Ablation Studies

To disentangle the contributions of the training stages and the ICC mechanism, we conduct an ablation study evaluating value function approximation performance and summarize the results in Table 2.

The Challenge of Zero-Shot Value Estimation. A critical observation is that even GPT-5, the oracle model we use to synthesize reasoning traces, struggles to provide accurate value estimates by prompting alone, standing in sharp contrast to the success of LLM-as-a-Judge in reward modeling. We attribute this discrepancy to the fundamental requirement of policy-conditioning. Without explicit training, even a strong generalist model fails to accurately predict the value of a specific policy, highlighting the necessity of our specialized training pipeline.

Impact of Training Stages. While SFT provides a large improvement over the base model by instilling the reasoning patterns needed for value prediction within a controlled token budget, its predictions are bounded by the limited accuracy of the oracle itself. RL pretraining is therefore necessary, as it enables continuous improvement by optimizing directly toward observed returns. We further note that the main benefit of the SFT stage is yielding a stable reasoning length rather than improving prediction accuracy, which RL pretraining alone is sufficient to improve (Appendix D). Thus the SFT stage is beneficial but not strictly necessary.

Table 2: Ablation on training pipeline and ICC in approximation performance (MSE).

Model	w/ ICC	w/o ICC
GPT-5	0.129	0.164
Qwen3-8B-Base	0.299	0.308
+ *SFT*	0.137	0.164
+ *RL*	0.051	0.058

Table 3: Relative ranking performance of different critics in top-1 accuracy.

# Candidates	Discriminative	Generative
2	0.604	0.711
4	0.388	0.492
8	0.204	0.332

Table 4: Generalization performance of generative critics over discriminative critics, measured by the approximation MSE reduction on datasets of varying distribution shift.

Dataset	Distribution Shift	Discriminative	Generative	Reduction
DeepScaleR	None	0.0666	0.0531	20.3%
Math500	Mild	0.1311	0.0796	39.3%
AIME24	Moderate	0.0524	0.0249	52.5%
GPQA	High	0.0583	0.0300	48.5%

Efficacy of ICC. We observe that removing ICC consistently degrades performance across all configurations. Although ICC alone is insufficient to transform a generalist LLM into an accurate value model, this consistent benefit indicates that it provides additional information that the model weights alone cannot fully capture, effectively enhancing the precision of value estimation.

5.3.2 Relative Ranking Performance

While approximation error measures the absolute performance of value models, RL optimization depends heavily on relative comparisons due to mechanisms like advantage estimation and group sampling. Therefore, assessing value prediction quality requires a relative ranking probe.

Setup. To evaluate this capability, we generate multiple candidate segments per prompt, compute their ground-truth values using the average score of Monte Carlo rollouts, and score them with either discriminative or generative critic. We measure the top-1 accuracy, i.e. the ratio of prompts where the critic's highest-scoring candidate matches the candidate with the highest ground-truth value, and report the results in Table 3.

Results. The results demonstrate that the generative critic consistently outperforms its discriminative counterpart across all candidate pool sizes. Crucially, the performance gap widens as the decision space grows larger: the discriminative critic's accuracy degrades quickly towards random chance while the generative critic maintains a much higher accuracy. This indicates that in addition to yielding better absolute approximation, the generative critic provides a more reliable advantage signal for distinguishing between possible actions.

5.3.3 Generalization

Although value models are mostly responsible for providing accurate feedback on in-distribution training data, their out-of-distribution generalization is crucial for effective and stable learning on exploratory off-policy data. A critic that overfits superficial in-distribution patterns will provide noisy or degenerate feedback that hinders the learning process and could lead to premature convergence.

Setup. To assess this robustness, we apply our approximation performance evaluation to the pretrained critics across a spectrum of datasets with increasing degrees of distributional shift. Concretely, we progressively move from exact in-distribution training (DeepScaleR) to mathematical problems of varying difficulty (Math500 for easier and AIME24 for harder), and finally to a distinct domain of general scientific reasoning (GPQA; Rein et al., 2024).

Results. The evaluation in Table 4 reveals that the two architectures differ markedly in generalization. While the generative critic achieves a modest MSE reduction on in-distribution data compared to

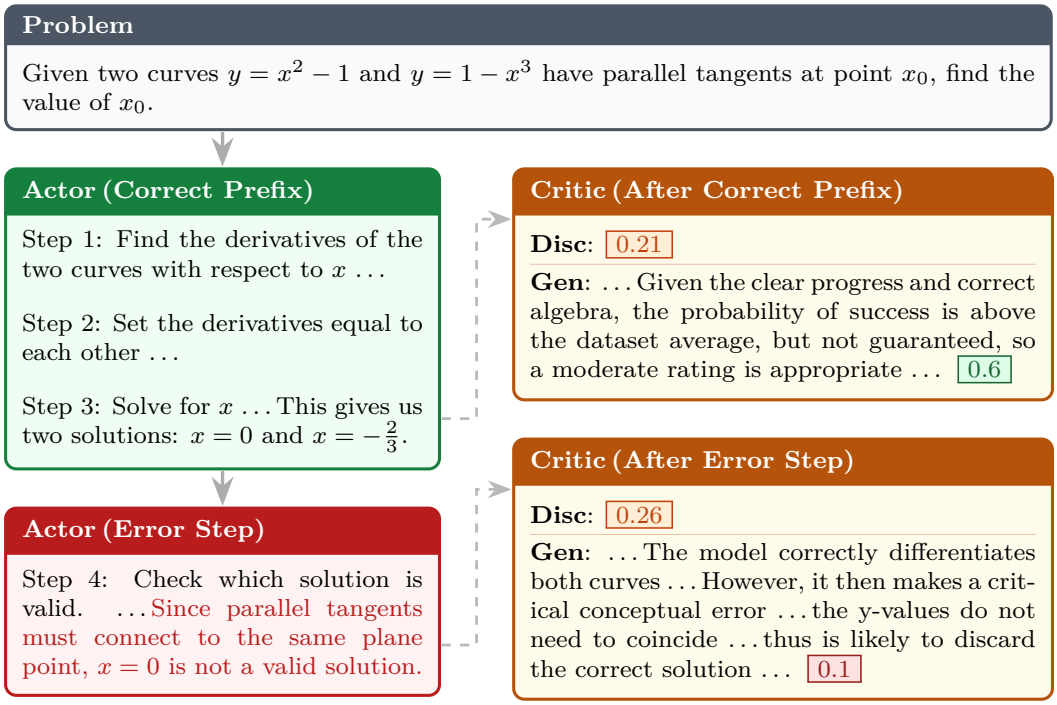


Figure 4: A case where the generative critic's accurate value estimates translate into detecting a conceptual error, while the discriminative critic provides less informative and misleading assessments.

the discriminative baseline, its relative advantage increases significantly as the distributional shift increases, reaching over 50% reduction on AIME24. This suggests that the generative critic evaluates reasoning quality based on transferable patterns rather than distribution-specific memorization. This stronger generalization helps explain the growing performance gap between GenAC and VC-PPO in Figure 2(b): as the actor evolves away from its initialization, the generative critic continues to provide reliable feedback while the discriminative critic degrades.

5.3.4 Case Study

We illustrate a qualitative case study in Figure 4. The problem requires finding the x-coordinate at which two curves possess parallel tangents. Following a correct prefix, the generative critic acknowledges this progress by assigning a value of 0.6. In Step 4, however, the actor commits a conceptual error by dismissing a valid solution. The generative critic accurately points out this misconception and assigns a low value of 0.1. Notably, since these predictions are independent, the abrupt value drop arises from two precise assessments. In contrast, the discriminative critic yields less informative and misleading predictions. This example demonstrates the generative critic's ability to assign credit both accurately and interpretably.

6 Conclusion and Limitations

This work improves credit assignment in LLM RL through generative value modeling. Our generative critics are more scalable, robust and accurate than discriminative critics; integrating them into PPO improves sample efficiency and final performance. Our limitations concern cost and the theory-practice gap. We focus on demonstrating what is possible with accurate credit assignment, providing techniques to reduce cost but leaving optimal cost tuning as future work. Our representation-complexity analysis motivates the architecture but does not guarantee an advantage under practical reasoning budgets. Overall, our results suggest that value modeling is a promising direction for improving credit assignment in LLM reinforcement learning.

AI USE STATEMENT

We used generative AI tools to polish writing and generate synthetic data for the critic's supervised fine-tuning stage, as described in Section 4.2 and Appendix C. The authors manually checked the polished paragraphs and take responsibility for the paper's final content.

ETHICS STATEMENT

This work studies value modeling for reinforcement learning of language models and does not involve human participants or crowdsourcing. Improved credit assignment may support more capable reasoning models, but such capabilities can also be applied to harmful objectives. We encourage responsible use in downstream applications.

REPRODUCIBILITY STATEMENT

Due to company compliance requirements, we are unable to release our source code, datasets or checkpoints. To support reproduction, we describe the model and training procedure in Section 4 and the experimental setup and evaluation in Section 5. Appendix C provides pseudocode, dataset construction and implementation details. We report multi-seed approximation experiments in Figure 2(a) and RL runs in Appendix E, and additional SFT ablations and computational cost analysis in Appendices D and F.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Jiaqi Chen, Bang Zhang, Ruotian Ma, Peisong Wang, Xiaodan Liang, Zhaopeng Tu, Xiaolong Li, and Kwan-Yee K Wong. Spc: Evolving self-play critic via adversarial games for llm reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, et al. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*, 2025b.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. In *International Conference on Machine Learning*, pp. 10818–10838. PMLR, 2025.

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

Tiantian Fan, Lingjun Liu, Yu Yue, Jiaze Chen, Chengyi Wang, Qiying Yu, Chi Zhang, Zhiqi Lin, Ruofei Zhu, Yufeng Yuan, et al. Truncated proximal policy optimization. *arXiv preprint arXiv:2506.15050*, 2025.

Guhao Feng and Han Zhong. Rethinking model-based, policy-based, and value-based reinforcement learning via the lens of representation complexity. *Advances in Neural Information Processing Systems*, 37:8569–8611, 2024.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information Processing Systems*, 36:70757–70798, 2023.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Xue Gong, Qi Yi, Ziyuan Nan, Guanhua Huang, Kejiao Li, Yuhao Jiang, Ruibin Xiong, Zenan Xu, Jiaming Guo, Shaohui Peng, et al. Segmental advantage estimation: Enhancing ppo for long-context llm training. *arXiv preprint arXiv:2601.07320*, 2026.

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

Jiaxin Guo, Zewen Chi, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. Reward reasoning models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b.

Yiran Guo, Lijie Xu, Jie Liu, Ye Dan, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in rl for large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025c.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 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.

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.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. In *International Conference on Machine Learning*, pp. 29557–29590. PMLR, 2025.

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. Process reward models that think. *arXiv preprint arXiv:2504.16828*, 2025.

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

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.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. *Advances in neural information processing systems*, 20, 2007.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35:21314–21328, 2022.

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.

Yingru Li, Jiawei Xu, Ziniu Li, Jiacai Liu, Wei Liu, Yuxuan Tong, Longtao Zheng, Zhenghai Xue, Yaxiang Zhang, Tianle Cai, et al. The optimal token baseline: Variance reduction for long-horizon llm-rl. *arXiv preprint arXiv:2602.07078*, 2026.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers transformers to solve inherently serial problems. In *The Twelfth International Conference on Learning Representations*, 2024.

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 *The twelfth international conference on learning representations*, 2023.

Jiashun Liu, Johan Obando-Ceron, Han Lu, Yancheng He, Weixun Wang, Wenbo Su, Bo Zheng, Pablo Samuel Castro, Aaron Courville, and Ling Pan. Asymmetric proximal policy optimization: mini-critics boost llm reasoning. *arXiv preprint arXiv:2510.01656*, 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.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, et al. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. `https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2`, 2025. Notion Blog.

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.

William Merrill and Ashish Sabharwal. The parallelism tradeoff: Limitations of log-precision transformers. *Transactions of the Association for Computational Linguistics*, 11:531–545, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement finetuning. In *International Conference on Machine Learning*, pp. 50893–50925. PMLR, 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. URL `https://openreview.net/forum?id=Ti67584b98`.

Kusha Sareen, Morgane M Moss, Alessandro Sordoni, Arian Hosseini, and Rishabh Agarwal. Putting the value back in rl: Better test-time scaling by unifying llm reasoners with verifiers. In *Second Conference on Language Modeling*, 2025.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, MA, 1998.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of llms should leverage suboptimal, on-policy data. In *International Conference on Machine Learning*, pp. 47441–47474. PMLR, 2024.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696.

Zhihui Xie, Jie Chen, Liyu Chen, Weichao Mao, Jingjing Xu, and Lingpeng Kong. Teaching language models to critique via reinforcement learning. In *International Conference on Machine Learning*, pp. 68559–68577. PMLR, 2025.

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

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. In *International Conference on Machine Learning*, pp. 73511–73525. PMLR, 2025a.

Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. What's behind ppo's collapse in long-cot? value optimization holds the secret. *arXiv preprint arXiv:2503.01491*, 2025b.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Kaiwen Zha, Zhengqi Gao, Maohao Shen, Zhang-Wei Hong, Duane S Boning, and Dina Katabi. Rl tango: Reinforcing generator and verifier together for language reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *The Thirteenth International Conference on Learning Representations*, 2024.

Yi-Kai Zhang, Zhiyuan Yao, Hongyan Hao, Yueqing Sun, Qi Gu, Hui Su, Xunliang Cai, De-Chuan Zhan, and Han-Jia Ye. v_0: A generalist value model for any policy at state zero. *arXiv preprint arXiv:2602.03584*, 2026.

Jian Zhao, Runze Liu, Kaiyan Zhang, Zhimu Zhou, Junqi Gao, Dong Li, Jiafei Lyu, Zhouyi Qian, Biqing Qi, Xiu Li, et al. Genprm: Scaling test-time compute of process reward models via generative reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34932–34940, 2026. doi: 10.1609/aaai.v40i41.40797.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. PyTorch FSDP: Experiences on scaling fully sharded data parallel. *Proceedings of the VLDB Endowment*, 16(12):3848–3860, 2023. doi: 10.14778/3611540.3611569.

Han Zhong, Zikang Shan, Guhao Feng, Wei Xiong, Xinle Cheng, Li Zhao, Di He, Jiang Bian, and Liwei Wang. Dpo meets ppo: Reinforced token optimization for rlhf. In *International Conference on Machine Learning*, pp. 78498–78521. PMLR, 2025a.

Han Zhong, Yutong Yin, Shenao Zhang, Xiaojun Xu, Yuanxin Liu, Yifei Zuo, Zhihan Liu, Boyi Liu, Sirui Zheng, Hongyi Guo, et al. Brite: Bootstrapping reinforced thinking process to enhance language model reasoning. *arXiv preprint arXiv:2501.18858*, 2025b.

Dingwei Zhu, Shihan Dou, Zhiheng Xi, Senjie Jin, Guoqiang Zhang, Jiazheng Zhang, Junjie Ye, Mingxu Chai, Enyu Zhou, Ming Zhang, et al. Vrpo: Rethinking value modeling for robust rl training under noisy supervision. *arXiv preprint arXiv:2508.03058*, 2025.

A Full Related Works

Value-Based Algorithms. Although crucial for fine-grained advantage estimation, value modeling remains underexplored beyond standard PPO in LLM RL. VC-PPO (Yuan et al., 2025b) mitigates initialization bias and reward decay through value pretraining and setting $\lambda = 1$ in TD(λ) for value learning. We adopt these techniques, but find them insufficient for effective value learning. VRPO (Zhu et al., 2025) introduces a complex information bottleneck architecture to handle noisy supervision; SAE (Gong et al., 2026) formulates a segment-level MDP to apply PPO on. However, all these works rely on discriminative architecture and do not isolate value function approximation as an independent factor of performance. Meanwhile, several works propose better ways of utilizing learned value models for effective policy learning (Yue et al., 2025; Fan et al., 2025; Liu et al., 2025a) instead of improving value modeling. Some work (Xie et al., 2025; Chen et al., 2025a) involves textual critique rationales instead of value estimates; V_0 (Zhang et al., 2026) learns a value model applicable only at the initial state. These works are orthogonal to ours.

Value-Free Algorithms. Bandit-style algorithms such as GRPO (Shao et al., 2024) and RLOO (Ahmadian et al., 2024) have gained popularity due to their simplicity and efficiency. While these methods avoid the cost of value modeling, they risk over-simplification and could lower the possible gains of post-training. To obtain fine-grained feedback without value models, methods like VinePPO (Kazemnejad et al., 2025) and SPO (Guo et al., 2025c) use multiple continuous rollouts for unbiased segment-level value estimation. However, these sample-based estimations incur a great inference cost during training. GiGPO (Feng et al., 2025) estimates advantage by utilizing repeated states that exist only in multi-turn agent training rather than general LLM RL. Recently, OTB (Li et al., 2026) derives a token baseline to approximate directly.

Fine-Grained Reward Signals. Another line of work tackles credit assignment by augmenting sparse outcome rewards with dense, intermediate reward signals. Process Reward Models (PRMs) provide reward at every intermediate reasoning step, requiring expensive process labels annotated by humans or rollouts (Lightman et al., 2023; Wang et al., 2024; Luo et al., 2024). Subsequent work proposes implicit PRMs that require only outcome labels to train (Zhong et al., 2025a;b; Yuan et al., 2025a; Cui et al., 2025). However, both explicit and implicit PRMs are static evaluators highly susceptible to reward hacking and often struggle to generalize. Other approaches design heuristic reward bonuses (Qu et al., 2025). These works reformulate the RL objective with external rewards, while our work focuses on utilizing internal feedback through value estimates.

Generative Reward Models. There is a growing interest in utilizing generative architectures for reward modeling. Early methods finetune language models to generate chain-of-thought reasoning before extracting probabilities from indicator tokens (Zhang et al., 2024; Mahan et al., 2024), while more recent works train models with RL to directly output verbalized rewards (Liu et al., 2025b; Chen et al., 2025b; Guo et al., 2025b; Sareen et al., 2025). Similar ideas are also explored in PRMs (Zhao et al., 2026; Khalifa et al., 2025; Zha et al., 2025). Reward models are not relevant as they predict sparse rewards without concerning credit assignment. Compared with process rewards, value modeling is a better way to implement credit assignment: value estimates function as baselines and don't change the optimized objective in expectation, making value modeling resistant to reward hacking and robust to imperfect models. In addition, value functions are defined over evolving policies, which reward functions aren't. We adapt the chain-of-thought generation to the more promising and difficult value modeling task, and our approach ensures effective and grounded value learning throughout the RL process.

B Background on Representation Complexity

Complexity Classes TC^0 and P. The complexity class P denotes problems solvable in polynomial time, which may still require many sequential computational steps. In contrast, TC^0 denotes computations realizable by polynomial-size, constant-depth threshold circuits, which are far more parallel and far less sequential. In particular, prior work shows that log-precision Transformers with constant depth and polynomial hidden dimension are confined to the complexity class TC^0 (Merrill & Sabharwal, 2023). The widely held assumption $\mathsf{TC}^0 \neq \mathsf{P}$ therefore says that some polynomial-time computations cannot be compressed into a constant number of layers.

Algorithm 1 GenAC

Require: Actor π_θ, generative critic V_ϕ, prompt dataset $\mathcal{D}$, actor verifier R, critic verifier R_v, ICC update momentum c, batch size B, minibatch size b

1: Initialize ICC hint $\bar{r} \leftarrow 0$
2: **for** each training iteration **do**
3: Sample a batch of prompts $\{x^i\}_{i=1}^{B} \sim \mathcal{D}$
4: **for** each prompt x^i **do**
5: Sample response $y^i \sim \pi_\theta(\cdot|x^i)$
6: Compute reward $r^i \leftarrow R(x^i, y^i)$
7: Update ICC hint $\bar{r} \leftarrow c \cdot \bar{r} + (1-c) \cdot r^i$
8: Compute old probabilities $\pi_{\text{old}}(y^i|x^i) \leftarrow \pi_\theta(y^i|x^i)$
9: **end for**
10: **for** each minibatch $\{(x^j, y^j, r^j, \pi_{\text{old}}(y^j|x^j))\}_{j\in[b]}$ **do**
11: **for** $j = 1, 2, \ldots, b$ **do**
12: Split response into segments $\{y_k^j\}_{k=1}^{T_j} \leftarrow \text{Segment}(y^j)$
13: **for** $k = 1, 2, \ldots, T_j$ **do**
14: Apply ICC prompt template $s_k^j \leftarrow \text{Template}(x^j, y_{:k}^j, \bar{r})$
15: Sample response from critic $z_k^j \sim V_\phi(\cdot|s_k^j)$
16: Parse value prediction $\hat{v}_k^j \leftarrow \text{Parse}(z_k^j)$
17: Compute critic reward $r_k^j \leftarrow R_v(z_k^j, r^j)$
18: **end for**
19: Broadcast to token value

$$\{\hat{v}_t^j\}_{t=1}^{|y^j|} \leftarrow \text{Broadcast}\big([\hat{v}_1^j, \hat{v}_2^j, \ldots, \hat{v}_{T_j}^j], [|y_1^j|, |y_2^j|, \ldots, |y_{T_j}^j|]\big)$$

20: Compute actor advantage $\{\hat{A}_t^j\}_{t=1}^{|y^j|} \leftarrow \text{BatchNorm}\left(\text{GAE}\left(r^j, \{\hat{v}_t^j\}_{t=1}^{|y^j|}\right)\right)$
21: **end for**
22: Update actor with PPO:

$$\mathcal{L}_\theta \leftarrow -\sum_{j=1}^{b}\sum_{t=1}^{|y^j|} \min\left(\frac{\pi_\theta(y_t^j|x^j \circ y_{<t}^j)}{\pi_{\text{old}}(y_t^j|x^j \circ y_{<t}^j)}\hat{A}_t^j, \text{clip}(\frac{\pi_\theta(y_t^j|x^j \circ y_{<t}^j)}{\pi_{\text{old}}(y_t^j|x^j \circ y_{<t}^j)}, 1-\epsilon, 1+\epsilon)\hat{A}_t^j\right)$$

23: Update critic with REINFORCE:

$$\mathcal{L}_\phi \leftarrow -\sum_{j=1}^{b}\sum_{k=1}^{T_j}\sum_{t=1}^{|z_k^j|} \log V_\phi(z_{k,t}^j|s_k^j \circ z_{k,<t}^j) r_k^j$$

24: **end for**
25: **end for**

Value-Focused Restatement from Feng & Zhong (2024). A value-focused restatement of their result is as follows. They construct a family of P MDPs indexed by an underlying language $L \in \mathsf{P}$, so the result applies to a broad class of constructions rather than a single example. When L is P-complete, computing the optimal value function for the corresponding MDP family is itself P-complete. In other words, even for an explicitly specified finite MDP, evaluating the correct value at a state can already capture the full difficulty of arbitrary polynomial-time computation. Their expressiveness result makes the implication for standard neural predictors more concrete: under the assumption $\mathsf{TC}^0 \neq \mathsf{P}$, these value functions cannot be represented by constant-depth, polynomial-width MLPs or Transformers. The takeaway is that value computation may intrinsically require more sequential reasoning than a shallow one-shot predictor can express.

C Implementation Details

Pseudocode. The pseudocode for GenAC joint training is presented in Algorithm 1. In line 12, the actor's response is segmented into a sequence of short segments using heuristic rules detailed below. Since group sampling is applied, sample-based initial values are available, critics are not queried or

Table 5: Hyperparameters used for RL Training.

Hyperparameter	Value
Actor Learning Rate	1e-6
Discriminative Critic Learning Rate	5e-6
Generative Critic Learning Rate	1e-6
Adam Betas	$(0.9, 0.999)$
Adam Weight Decay	0.01
Training Steps	200
Prompt Batch Size	128
Group Size	8
Micro Batch Size	256
Maximum Prompt Length	1024
Maximum Response Length	8192
Maximum Critic Response Length	1024
PPO ϵ	0.2
GAE λ	1.0
Discount γ	1.0

evaluated for predictions on initial states. In line 19, the resulting segment-level value predictions are broadcast back to token-level to match the token-level objective in standard PPO. In line 20, following standard practice, we use GAE with $\lambda = 1$ and batch normalization to compute advantage estimates.

Segmentation Rule. We employ a delimiter-based heuristic to decompose a full model response into coherent segments. Specifically, we first identify candidate delimiters (periods and newline characters) and filter out any that fall within mathematical expressions. The remaining delimiters are used as segmentation boundaries. In addition, we then merge consecutive short segments until each meets a minimum length of 64 tokens, yielding an average segment length of approximately 120 tokens.

Code. We develop our code based on verl (Sheng et al., 2024) with vLLM (Kwon et al., 2023) engine and FSDP (Zhao et al., 2023) backend. All experiments are conducted on a node of Nvidia 80G A100 GPUs.

Dataset Construction. All training and validation datasets are derived from the DeepScaleR dataset. The value benchmark in the approximation experiments includes 1024 problems, each with one response split into segments to construct 14190 states, and each state conducts 16 partial rollouts to obtain value estimate used as ground-truth target. The RL pretrain prompt dataset includes 12800 problems, or 100 RL steps. This dataset is also used as the SFT problem set. For each problem and one of its responses, we randomly select one prefix, construct the prompt with template shown in Figure 3, query GPT-5, then collect model outputs without further processing. The main RL prompt dataset includes 25600 problems, or 200 RL steps. The relative ranking experiment involves 1024 problems, each 8 candidates, whose ground-truth value is estimated through 32 partial rollouts. The generalization experiment constructs value benchmarks of different sources similarly as the DeepScaleR dataset, with an exception of using multiple responses per problem (2 for Math500, 30 for AIME24, 4 for GPQA) to roughly match the amount of total sequences.

Hyperparameters for the RL Experiment. All hyperparameters are detailed in Table 5, largely following the example scripts in verl. During validation, we sample 16 responses per problem using the same sampling parameters as during training and report their average accuracy.

Hyperparameters for the Approximation Experiment. For discriminative critics, we perform a hyperparameter search over learning rates in $\{1e-5, 5e-6, 4e-6, 2e-6, 1e-6\}$ and training steps in $\{100, 200\}$ per model size. For generative critics, we fix the learning rate to $1e-6$ and the number of training steps to 100, as they are much more robust to such variation. See Table 6 for the final selected hyperparameters.

Table 6: Hyperparameters used for approximation scaling experiments.

Model Size	Discriminative Critics		Generative Critics	
	Learning Rate	Training Steps	Learning Rate	Training Steps
0.6	1e-5	100		
1.7	5e-6	100		
4	4e-6	200	1e-6	100
8	1e-6	200		
14	1e-6	200		

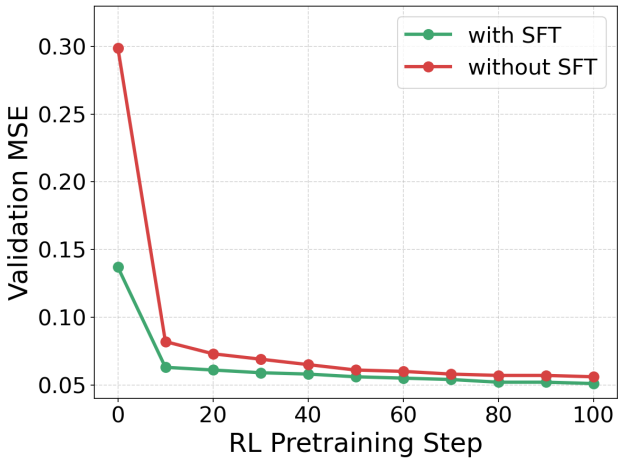


Figure 5: Validation MSE of generative critics during RL pretraining, with and without the SFT stage.

D Ablation of the SFT Stage

Setup. To isolate the effect of the SFT stage, we skip it entirely, initializing the critic directly from the base model, and run the identical RL pretraining procedure. We report the validation MSE in Figure 5.

Results. RL pretraining alone recovers most of the MSE reduction, reaching 0.056 at the end of training versus 0.051 with SFT. The remaining difference lies in the stability of the critic's reasoning length: without SFT the chain-of-thought length starts longer and fluctuates substantially across training, whereas with SFT it stays within a narrow band. The SFT stage therefore serves as a natural warmup that provides a stable reasoning format and token budget, while RL pretraining is responsible for the accuracy, suggesting that in our pipeline, it is beneficial rather than necessary.

E Multi-Seed Robustness

Setup. We repeat the GRPO and VC-PPO with two additional random seeds, in total three runs per baseline, and report all results in Table 7.

Results. Both GRPO and VC-PPO perform very similarly across all three runs, with average scores stable to within approximately 0.4 points. This low run-to-run variance also indicates the robustness of both the algorithms and our evaluation protocol. GenAC is clearly distinguished from both.

Table 7: Final performance in `avg@16` accuracy (%) on validation benchmarks, including two additional runs of GRPO and VC-PPO with different random seeds.

	AIME24	AMC23	MATH500	AIME25	Olympiad	Minerva	Avg
GRPO (Table 1)	26.46	69.69	87.20	22.92	52.92	42.12	50.22
GRPO (seed 2)	24.17	69.06	87.74	22.71	53.06	44.23	50.16
GRPO (seed 3)	24.58	68.28	87.09	21.25	53.23	42.56	49.50
GRPO (mean)	25.07	69.01	87.34	22.29	53.07	42.97	49.96
RLOO (Table 1)	24.79	68.75	87.64	21.25	53.94	43.22	49.93
VC-PPO (Table 1)	25.83	70.78	87.54	21.67	53.50	43.02	50.39
VC-PPO (seed 2)	27.50	70.78	87.94	22.71	53.86	44.58	51.23
VC-PPO (seed 3)	25.21	72.34	88.46	21.46	54.23	43.87	50.93
VC-PPO (mean)	26.18	71.30	87.98	21.95	53.86	43.82	50.85
GenAC (Table 1)	30.00	74.53	87.48	21.88	54.16	43.34	51.90

F Computational Cost Analysis of GenAC

We analyze the total FLOPs of GenAC in comparison to GRPO, VC-PPO and VinePPO. We begin by deriving per-step cost, and then total cost including initialization, before discussing immediate cost reduction implementations under our framework.

Per-Iteration Costs. Following standard practice, we adopt the following formula to count FLOPs:

$$C_{\text{generation}} = 2PT$$
$$C_{\text{forward}} = 2PT$$
$$C_{\text{train}} = 6PT$$

where P is the number of model parameters and T is the number of processed tokens.

The actor terms are common to all methods: $2P_{\text{A}}T_{\text{A}}$ for generation, $2P_{\text{A}}T_{\text{A}}$ for the old logprob forward pass and $6P_{\text{A}}T_{\text{A}}$ for training. GRPO uses this alone:

$$C_{\text{GRPO}} = 2P_{\text{A}}T_{\text{A}} + 2P_{\text{A}}T_{\text{A}} + 6P_{\text{A}}T_{\text{A}} = 10P_{\text{A}}T_{\text{A}}.$$

PPO involves a discriminative critic of similar size to the actor, which reads the same tokens:

$$C_{\text{PPO}} = 10P_{\text{A}}T_{\text{A}} + 2P_{\text{C}}T_{\text{A}} + 6P_{\text{C}}T_{\text{A}} = 10P_{\text{A}}T_{\text{A}} + 8P_{\text{C}}T_{\text{A}}.$$

VC-PPO uses the same critic within the RL training and thus has the same per-iteration cost. GenAC utilizes a generative critic that generates and learns on different tokens. With average segment length L_1 and average critic generation length L_2, the critic processes the full actor trace plus branched generations,

$$T_{\text{GC}} = T_{\text{A}} + \frac{T_{\text{A}}}{L_1}L_2,$$
$$C_{\text{GenAC}} = 10P_{\text{A}}T_{\text{A}} + 2P_{\text{GC}}T_{\text{GC}} + 6P_{\text{GC}}T_{\text{GC}} = 10P_{\text{A}}T_{\text{A}} + 8P_{\text{GC}}T_{\text{GC}}.$$

VinePPO involves branched Monte Carlo rollouts. With average branch length L_3 and n branches per prefix,

$$T_{\text{vine}} = \frac{T_{\text{A}}}{L_1}L_3 n,$$
$$C_{\text{VinePPO}} = 10P_{\text{A}}T_{\text{A}} + 2P_{\text{A}}T_{\text{vine}}.$$

Total Cost. The main RL training lasts 200 steps, each 128 problems, 8 responses per problem, and average response length 2048 tokens (1000 at the initial stage, 4000 at late stage):

$$T_{\text{A}} \approx 200 \times 128 \times 8 \times 2048 = 419.4\text{M}.$$

Table 8: Summary of per-iteration relative cost and total cost including one-time initialization. GenAC's cost is intentionally untuned, and it can be reduced substantially under our framework.

Method	Per iteration	Total FLOPs (omit P)
GRPO	$0.6\times$	4194M
PPO	$1.0\times$	7549.2M
VC-PPO	$1.0\times$	8778M
GenAC	$2.1\times$	18866M
VinePPO	$4.3\times$	32825M

With $L_1 \approx 120$, $L_2 \approx 300$, $L_3 \approx 1024$ (half of the average response length) and $n = 4$:

$$T_{\mathrm{GC}} = T_{\mathrm{A}} + \frac{T_{\mathrm{A}}}{L_1} L_2 \approx 1467.9\mathrm{M},$$
$$T_{\mathrm{vine}} = \frac{T_{\mathrm{A}}}{L_1} L_3 n \approx 14315.5\mathrm{M}.$$

The discriminative value pretraining uses 200 steps and average initial response length of 1000 tokens:

$$T_{\mathrm{VPD}} \approx 128 \times 8 \times 200 \times 1000 = 204.8\mathrm{M}.$$

SFT uses total trajectory count 12800, average response length 1000 tokens, and average critic length 300 tokens:

$$T_{\mathrm{SFT}} \approx 12800 \times \left(\frac{1000}{2} + 300 \right) = 10.2\mathrm{M}.$$

The generative value pretraining uses 100 steps, average initial response length 1000 tokens, average segment length of 120 tokens, and an average critic length of 300 tokens:

$$T_{\mathrm{VPG}} \approx 128 \times 8 \times 100 \times \left(1000 + \frac{1000}{120} \times 300 \right) = 358.4\mathrm{M}.$$

Plugging in and omitting $P_{\mathrm{A}} \approx P_{\mathrm{C}} \approx P_{\mathrm{GC}}$, the total cost of each method, including initialization, is

$$\begin{aligned}
C_{\mathrm{GRPO}} &= 10 T_{\mathrm{A}} \approx 4194\mathrm{M}, \\
C_{\mathrm{PPO}} &\approx 18 T_{\mathrm{A}} \approx 7549.2\mathrm{M}, \\
C_{\text{VC-PPO}} &\approx 6 T_{\mathrm{VPD}} + 18 T_{\mathrm{A}} \approx 1228.8\mathrm{M} + 7549.2\mathrm{M} = 8778\mathrm{M}, \\
C_{\mathrm{GenAC}} &\approx 6 T_{\mathrm{SFT}} + 8 T_{\mathrm{VPG}} + 10 T_{\mathrm{A}} + 8 T_{\mathrm{GC}} \\
&\approx 61.2\mathrm{M} + 2867.2\mathrm{M} + 4194\mathrm{M} + 11743.2\mathrm{M} \\
&= 18866\mathrm{M}, \\
C_{\mathrm{VinePPO}} &= 10 T_{\mathrm{A}} + 2 T_{\mathrm{vine}} \approx 4194\mathrm{M} + 28631\mathrm{M} = 32825\mathrm{M}.
\end{aligned}$$

Initialization accounts for 14% of VC-PPO's total cost and 16% of GenAC's. Both the per-step and total cost are summarized in Table 8. We note that we choose not to tune GenAC's cost, which can be heavily reduced within our framework as discussed below.

Reducing the Cost. The one-time initialization cost can be reduced as the SFT stage can be skipped and the value pretraining MSE curve saturates early (Appendix D), so both can be shortened with little performance loss. The dominant term in RL training is the generative critic inference and training cost ($8PT_{\mathrm{GC}} \approx 2.8\times$ GRPO) that depends on two hyperparameters: the segmentation strategy, affecting the average segment length (L_1), and the critic decode budget, affecting the average critic response length (L_2). Our deliberate choice of near-maximal feedback density (effectively about 33 segments per 4000-token response at late RL training) makes T_{GC} high. With coarser segmentation, the cost can be significantly reduced (Table 9).

In addition, generation at fine granularity is required for dense actor feedback, but training the critic on every generated sequence is a choice rather than a requirement. Our implementation fixes the number of critic minibatches to keep off-policyness constant, absorbing the varying sequence count

Table 9: GenAC per-iteration cost under coarser feedback granularity.

# Feedback	$T_{\mathrm{GC}}/T_{\mathrm{A}}$	Relative cost
33	3.5	$3.8\times$
10	1.75	$2.4\times$
5	1.375	$2.1\times$
1	1.075	$1.9\times$

Table 10: GenAC per-iteration cost with subsampled critic training set.

# Feedback	# Critic batch size ($\times 256$)	Relative cost
33	10	$2.8\times$
33	5	$2.5\times$
33	1	$2.3\times$
10	5	$2.2\times$
10	1	$2.0\times$
5	1	$1.9\times$

into a minibatch size that easily reaches 256×33, over an order of magnitude above standard choices of 256. Therefore, subsampling the critic's training set while generating at full granularity preserves the actor feedback density while drastically cutting the cost (Table 10).

Both tables are arithmetic under our cost model, and the cost is adjustable along both axes, decreasing asymptotically to PPO.

Summary. Table 8 summarizes the per-iteration relative cost and the total cost including one-time initializations. Value-based methods are generally costlier than value-free alternatives, but this is compensated by increased sample efficiency and continued gains in the long run (Section 5.2). Additionally, the cost of GenAC is an implementation choice, as we choose to explore what is achievable with the strongest value model we can build. We discuss how to reduce cost within our framework, and leave further cost-optimal tuning to future work.