Let's Try Again: Eliciting Multi-Turn Reasoning in Language Models via Simplistic Feedback

Anonymous Author(s)
Affiliation
Address
email

Abstract

Multi-turn problem solving is a critical yet challenging scenario in the practical application of Large Reasoning Models (LRMs), commonly encountered in domains such as chatbots, programming assistants, and education. Recently, reasoning models like DeepSeek-R1 has shown the promise of reinforcement learning (RL) methods in enhancing model reasoning capabilities. However, we observe that models trained with existing single-turn RL paradigms often **lose their ability to solve problems across multiple turns**, struggling to revise answers based on context and exhibiting repetitive responses. This raises new challenges in preserving reasoning abilities while enabling multi-turn contextual adaptation. In this work, we find that simply allowing models to engage in multi-turn problem solving where they receive only unary feedback (e.g., "Let's try again") after incorrect answers can help recover both single-turn and interactive multi-turn reasoning skills. We introduce **Unary Feedback as Observation (UFO)** for reinforcement learning, a method that explicitly leverages minimal yet natural user feedback during iterative problem-solving which can be easily applied to any existing single-turn RL training paradigm. Experimental results show that RL training with UFO preserves single-turn performance while improving multi-turn reasoning accuracy by 14%, effectively utilizing sparse feedback signals when available. To further reduce superficial guessing and encourage comprehensive reasoning, we explore reward structures that incentivize thoughtful, deliberate answers across interaction turns. Code and models will be publicly released.

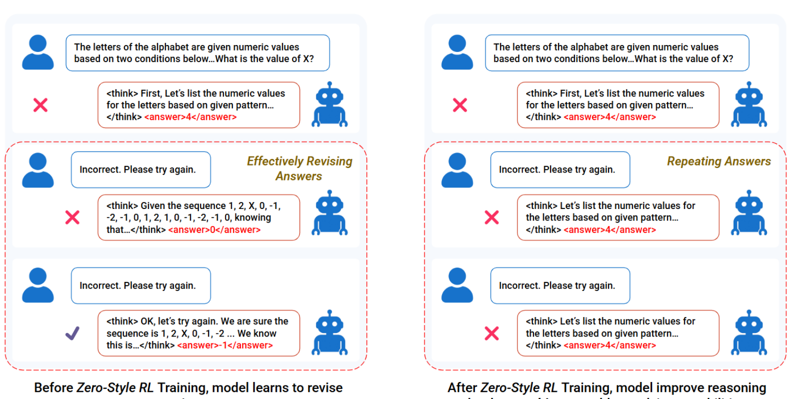

Figure 1: An example of using single-turn RL model for multi-turn problem solving. A single-turn RL trained model lose multi-turn capability, producing identical reasoning chains across interaction turns after being prompted that its answer is incorrect.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

1 Introduction

Large language and reasoning models (LLMs/LRMs) like DeepSeek-R1 [1–4] have shown promise in solving complex tasks such as mathematical problem solving and code generation. Multi-turn problem solving is particularly prevalent in real-world applications including chatbots, programming assistants, and educational tools, where users engage in interactive feedback loops to refine model responses [5–9]. Reinforcement Learning (RL) [10, 1, 11, 12] has become trending for LLM/LRM post-training, improving reasoning capabilities by rewarding correct model responses while penalizing incorrect ones. However, it remains underexplored whether these models trained with single-turn RL can generalize to real-world interactive problem-solving settings.

In this work, we first observe that single-turn RL may hinder a model's ability to engage in interactive, multi-turn reasoning. Specifically, such models often fail to incorporate in-context feedback and instead **persist with their initial answers across multiple turns** (Figure 1). Moreover, we find that in 70% of failed cases, single-turn-trained models generate exactly the same answer across five interaction turns (Figure 2). Though these models excel at single turns, how to make them effectively leverage in-context feedback and improve in multi-turn setting remains a challenge, especially considering current reasoning datasets are natually single-turn and lack multi-turn feedback incorporation which could also be very expensive. This gap motivates our central research question: *How can we train language models that not only generate correct solutions but also improve iteratively from sparse, minimal feedback?*

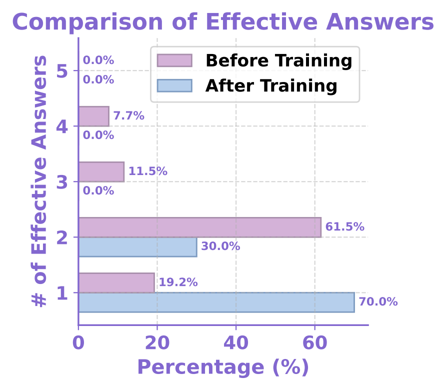

Figure 2: Validation comparison. Repetition rate remains high in later validation steps under single-turn RL.

Real-world multi-turn user feedback is very expensive and hard to obtain. Bottle-necked by this, existing multi-turn framework has been focusing on automatic feedback such as code interpreter messages [5, 6, 9] and embodied simulator signals [8, 13], while those inherently single-turn static dataset (e.g. QA, Math) are still predominantly leveraged for single-turn RL training. Moreover, code interpreter and embodied environment still requires large amount of resource and costly to build [14]. Considering all above, in this work we explore a surprisingly simple yet effective framework that could leverage static dataset for multi-turn RL training, by simply adding verbal unary feedback and encourage the model to *try again* when the model is wrong. We call this Unary Feedback as Observation (UFO), as it only occurs in context until model's answer is finally correct and the user ends the conversation without providing further signals.

Through experiments, we show that applying UFO in multi-turn RL settings [15] could effectively simulate interactive reasoning and allow the model to effectively revise its reasoning across turns. Models trained via UFO inherently learns to try to use different approaches when the current answer is wrong, and get significantly 14% higher success rates when evaluated under multi-turn scenarios compared to previous single-turn RL approaches. Interestingly, we find the model trained with UFO also exhibit higher performance in single-turn reasoning, suggesting that multi-turn training may promote better internalization of reasoning strategies, which generalize even in the absence of feedback. We presume this could be due to multi-turn setting inherently enable the model to explore diverse thinking patterns and learn to self reflection. We further investigated methods to amplify such self-reflection capabilities, and find that applying a turn-wise reward decay could encourage the model to think more systematically before answering, effectively enhancing reasoning capabilities and perform better with constrained interaction overhead.

To summarize, our contributions are as follows:

- We identify that while current single-turn RL training improves reasoning, they could lead to repetitive and degraded outputs in multi-turn interactive reasoning scenarios.
- We explore a simple yet effective framework, **Unary Feedback as Observation (UFO)**, to enable multi-turn RL training on existing static single-turn reasoning datasets.
- We show that UFO improves both multi-turn and single-turn reasoning, and that reward decay further enhances self-reflection and problem-solving behavior.

2 Reinforcement Learning for LLM Reasoning

2.1 Background

Single-Turn Reinforcement Learning. Reinforcement Learning (RL) is a standard way to align large language models (LLMs) with human preferences by maximizing

$$\mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_\theta(\cdot|x)}[R(x,y)],$$

where $\mathcal{D}$ is a prompt distribution, π_θ the policy, and $R(x,y)$ the reward for response y. Algorithms such as PPO [10, 16] and GRPO [1, 17] apply this objective to static datasets, yielding strong single-turn gains in math and code generation.

Multi-Turn Extensions. Real applications—tutoring, coding assistants, embodied agents—demand *multi-turn* interaction, where a model refines answers across steps under sparse feedback. RAGEN [12] addresses this by framing reasoning as an MDP and optimizing whole trajectories, supporting delayed credit assignment in tasks like symbolic logic and interactive programming. Yet widely used math and code datasets remain single-turn, and collecting turn-by-turn human signals is costly. Most prior work instead synthesizes feedback [6, 5] or uses tool-augmented environments [18, 19, 9], leaving open the key question we study here: *Can models trained only with single-turn RL generalize to multi-turn reasoning?*

2.2 Single-Turn RL Leads to Collapsed Multi-Turn Reasoning

To answer the question posed above, we examine how models trained with single-turn reinforcement learning (RL) perform in multi-turn interaction settings. While single-turn RL improves response quality in isolated prompts, we consistently observe that such models fail to revise their answers when provided with corrective feedback. In practical use cases such as tutoring or mathematical assistance, users typically offer minimal feedback (e.g., "try again") and expect the model to adjust its reasoning accordingly. However, RL-trained models often repeat their initial response even after being explicitly told it is incorrect, indicating a collapse in multi-turn reasoning capability.

This phenomenon is illustrated in Figure 1: a pre-trained model (left) gradually improves its response through contextual revision, while a single-turn RL model (right) produces the same output across turns, failing to incorporate feedback. We define an *effective answer* as a distinct attempt that is not a near-duplicate of any previous response within the same episode (e.g., 1.5 and 3/2 are considered duplicates). Figure 2 shows that after single-turn RL training, over 70% of model responses concentrate in the first attempt, with little variation in subsequent turns. We attribute this failure mode to two key factors: (1) the reward structure in single-turn RL offers no incentive for incremental improvement, and (2) the policy is trained without access to interaction history, making it insensitive to feedback. These limitations hinder the model's ability to develop revision-aware reasoning strategies.

This observation presents a central challenge: current single-turn RL frameworks are insufficient for acquiring multi-turn reasoning abilities, yet obtaining fine-grained, turn-level supervision in real-world tasks—especially in static domains like math—is prohibitively expensive.

In light of this, we ask the following question: **Can we leverage only the simplest form of supervision, such as "try again", to simulate multi-turn interaction on static datasets and train models to learn adaptive revision behaviors?**

Can minimal feedback alone unlock multi-turn reasoning on static datasets?

3 Training Multi-Turn Reasoning Models with Minimal Feedback

3.1 Problem Formulation

We model the process of multi-turn problem solving based on static single-turn datasets as a finite-horizon Markov Decision Process (MDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, T_{\text{max}})$. Here, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space consisting of all possible answers, $\mathcal{P}$ is the transition function defined by the agent–environment interaction, R is the reward function, and T_{max} is the maximum number of interaction steps per episode. At each turn t, the agent observes a state $s_t \in \mathcal{S}$ that encodes the original question q and the history of past attempts and feedbacks:

$$s_t = \texttt{Concat}(q, \{(a_k, f_k)\}_{k=1}^{t-1}), \tag{1}$$

where a_k denotes the k-th answer, and f_k is a binary feedback token returned by the environment. Importantly, the feedback in the observation is restricted to **negative signals** such as `TryAgain`. When the agent produces a correct answer, the episode terminates immediately, and no explicit confirmation (e.g., `Correct`) is added to the context. As a result, the agent only receives **unary feedback** and must learn to revise its answers based solely on a history of failed attempts. It generates an answer $a_t \sim \pi_\theta(\cdot \mid s_t)$ and receives a scalar reward:

$$r_t = \begin{cases} 1, & \text{if } a_t \text{ is correct,} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The episode ends when the agent provides a correct answer or reaches the maximum number of steps T_{max}.

This formulation reflects the sparse and delayed nature of real-world feedback in reasoning tasks. It does not rely on access to ground-truth reasoning traces or executable environments. Unlike conventional RL settings that assume dense or tool-driven feedback, our MDP is instantiated entirely from static, single-turn datasets and minimal correctness signals—making it well-suited for weakly-supervised multi-turn training.

3.2 Unary Feedback as Observation (UFO)

To enable multi-turn reasoning on static datasets without access to tools or step-by-step labels, we propose a simple yet general mechanism called **Unary Feedback as Observation (UFO)**. The key idea is to treat minimal feedback (specifically, failure-only textual signals) as part of the observation, allowing the model to revise its answers based on previously failed attempts.

At each turn t, the model receives an input constructed as:

$$x_t = \texttt{Format}(q, \{(a_k, f_k)\}_{k=1}^{t-1}), \tag{3}$$

where q is the original question, and each (a_k, f_k) represents a previous answer a_k and its associated feedback signal f_k. Since episodes terminate immediately after a correct answer, no positive signals (e.g., `Correct`) are ever included in the context. Thus, the agent must learn to revise its reasoning based solely on failure traces (e.g., $f_k \in \{\texttt{TryAgain}\}$).

In practice, the prompt is constructed as a natural-language sequence concatenating all previous attempts and their feedbacks. For example:

```
Question:  What is the value of ...?
Attempt 1:  [wrong answer]
Feedback:  Try Again.
...
Attempt K: [correct answer]
```

This formulation enables us to transform static single-turn datasets into multi-turn interaction episodes without requiring structural changes, expert annotations, or execution environments. Thus, UFO allows multi-turn reinforcement learning on LLMs with minimal supervision. We describe this training setup as follows.

3.3 Reinforcement Learning with Minimal Feedback

Given the MDP formulation and the UFO-based observation design, we optimize the agent using reinforcement learning to learn revision-aware, multi-turn policies. Since the dataset contains only final-answer correctness and lacks ground-truth reasoning traces, supervised finetuning is not applicable. Reinforcement learning, by contrast, enables exploration of diverse reasoning strategies under sparse and delayed supervision.

We adopt Proximal Policy Optimization (PPO) to train the policy π_θ, following prior work [20, 12] which shows that a learned critic enables fine-grained value estimates and stabilizes optimization.

At each episode, the agent interacts with a problem over multiple rounds. At each turn t, it observes input x_t, generates an answer a_t, and receives a binary reward $r_t \in \{0, 1\}$. The resulting trajectory is defined as:

$$\tau = \{(x_1, a_1, r_1), (x_2, a_2, r_2), \ldots, (x_T, a_T, r_T)\}, \tag{4}$$

where $T \leq T_{\text{max}}$ is the number of turns before success or termination. The objective is to maximize the expected return:

$$\mathcal{J}^{\text{RL}}(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=1}^{T} r_t\right]. \tag{5}$$

We apply PPO with a clipped surrogate objective. For each training batch, we estimate the advantage $\hat{A}_t$ using a baseline value function and update the policy as:

$$\mathcal{L}^{\text{PPO}}(\theta) = \mathbb{E}_t\left[\min\left(\frac{\pi_\theta(a_t \mid x_t)}{\pi_{\theta_{\text{old}}}(a_t \mid x_t)}\hat{A}_t,\ \text{clip}(\frac{\pi_\theta(a_t \mid x_t)}{\pi_{\theta_{\text{old}}}(a_t \mid x_t)}\hat{A}_t, 1-\epsilon, 1+\epsilon)\right)\right]. \tag{6}$$

Crucially, the UFO design enables the policy to condition on the full history of failure signals, giving rise to context-sensitive behaviors such as error correction, elimination, and hypothesis refinement—capabilities that are difficult to elicit through static supervision alone.

3.4 Reward Design for Adaptive Reasoning

Binary correctness signals offer a minimal form of supervision, but they often induce suboptimal behavior such as blind trial-and-error or repeated guesses. To encourage more efficient and reflective reasoning, we introduce two complementary reward shaping strategies: *reward decay* and a trajectory-level *repetition penalty*.

To promote early success, we apply exponential decay to the reward when a correct answer is produced at turn t:

$$\text{DecayReward}(t) = \begin{cases} \gamma^t, & \text{if } a_t \text{ is correct,} \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

where $\gamma \in (0, 1)$ is a decay factor that favors solving the problem in fewer turns.

To reduce answer repetition, we define a global penalty based on the number of *effective answers*—i.e., responses that are not near-duplicates of any prior attempt in the same episode. Let T denote the number of turns in the episode, and $E(\tau)$ the number of effective answers in the trajectory τ. We define a normalized penalty term:

$$\text{Penalty}(\tau) = \lambda \cdot \left(1 - \frac{E(\tau)}{T}\right), \tag{8}$$

where $\lambda > 0$ is a tunable penalty weight, and $E(\tau)/T$ measures answer diversity. The penalty reaches its maximum when all answers are identical, and disappears when all are distinct.

Combining these two components, the final reward used during training is defined as:

$$r_t = \begin{cases} \gamma^t - \lambda \cdot \left(1 - \frac{E(\tau)}{T}\right), & \text{if } a_t \text{ is correct,} \\ 0 - \lambda \cdot \left(1 - \frac{E(\tau)}{T}\right), & \text{otherwise.} \end{cases} \tag{9}$$

This formulation encourages both early success and multi-turn diversity, while requiring no additional supervision. In our experiments, we find that reward decay improves convergence and sample efficiency, whereas repetition penalties lead to more exploratory and reflective behaviors. Together, they significantly stabilize training in the minimal-feedback setting.

4 Experiments

4.1 Experimental Setup

Tasks and Environment Settings. All experiments are conducted on the MATH partition of the METAMATHQA dataset, where data are augmented from the training sets MATH. This environment provides math questions with adequate difficulty, enabling us to observe and analyze its reasoning emergence. We use an adapted version of the RAGEN [15] codebase which supports effective multi-turn RL training.

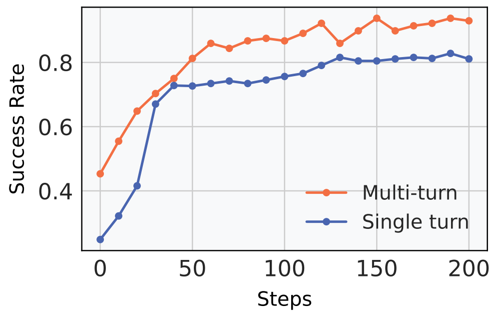

Figure 3: Multi-turn RL significantly outperforms single-turn baselines, achieving higher success rates with similar inference cost.

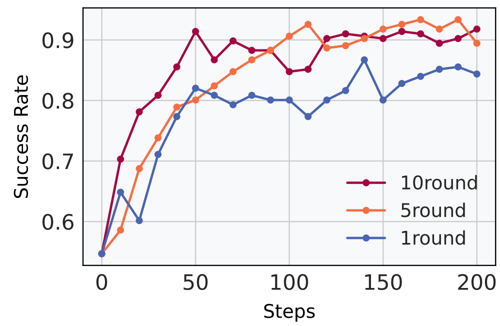

Figure 4: Moderate episodes (5 turns) for multi-turn training yield the best performance, while increasing to 10 turns offers no explicit gain.

Training Settings. We train Qwen-2.5-3B-Instruct with PPO for 200 optimization steps on A100 GPUs. Each batch samples $P{=}8$ prompts, with $N{=}16$ rollouts per prompt. During training, we experiment with three distinct configurations for the maximum number of turns per episode, setting $T_{\max}$ to 1, 5, and 10, respectively. For the validation phase, $T_{\max}$ is fixed at 5 turns. In both training and validation, episodes are limited to a maximum of 10 actions in total. Policy updates use PPO with GAE parameters $(\gamma, \lambda) = (1.0, 1.0)$, Adam with $\beta = (0.9, 0.999)$, entropy coefficient 10^{-3}.

Evaluation Metrics. We report two complementary metrics to assess both effectiveness and efficiency.

- **Success Rate (`Succ@k`).** This metric measures the percentage of problems solved within a fixed number of interaction turns. Let τ_j be the number of turns the agent takes to solve problem q_j, or ∞ if it fails. We have:

$$\text{Succ@k} = \frac{1}{N} \sum_{j=1}^{N} \mathbb{1}[\tau_j \leq k] \tag{10}$$

  We report `Succ@1` for single-turn performance, and `Succ@5/10` to reflect multi-turn capability.
- **Average Number of Turns.** To evaluate interaction efficiency, we report the average number of turns the agent takes to solve each problem:

$$\mathbb{E}[\texttt{num\_turns}] = \frac{1}{N} \sum_{j=1}^{N} T_j \tag{11}$$

  where T_j denotes the number of interactive turns taken for problem q_j. This metric reflects how efficiently the agent reaches a solution, accounting for retries and step-wise refinement across multi-turn episodes.

4.2 Experimental Results and Findings

In this section, we present empirical findings that address three central questions in our study of multi-turn reinforcement learning with unary feedback:

1. Section 4.2.1: Does multi-turn RL unlock stronger reasoning than single-turn training?
2. Section 4.2.2: Can models effectively revise their answers from sparse feedback alone?
3. Section 4.2.3: How do reward shaping strategies impact reasoning efficiency and diversity?

We explore each question in the following subsections, with quantitative analyses and ablation studies. Additional qualitative examples and robustness checks are included in the Supplementary.

4.2.1 Multi-turn RL Unlock Higher Upper Bound of LLM Reasoning

We compare models trained with multi-turn RL against single-turn PPO baselines, using `Succ@5` on a held-out validation set evaluated at 21 checkpoints across 200 training steps. During validation, each agent is allowed up to 5 interaction turns per problem ($k = 5$).

To ensure a fair comparison between single- and multi-turn methods, we evaluate the single-turn model by sampling 5 independent completions per problem, equivalent to `Pass@5`, while the

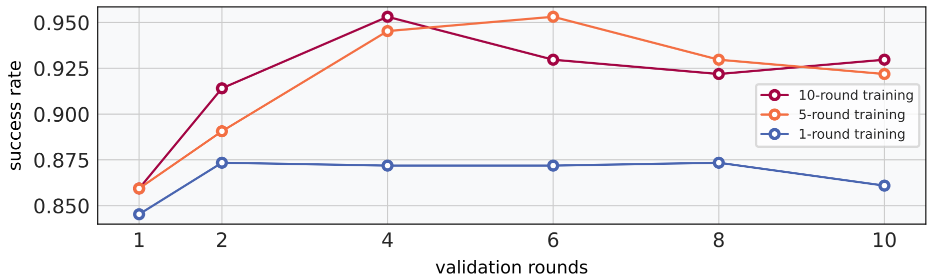

Figure 5: Validation performance (`Succ@k`) of models trained with different rollout turns under varying inference-time turn budgets. Multi-turn training (5 or 10 turns) consistently yields higher success rates across all inference turn budgets, including $k = 1$, indicating better generalization even to single-turn reasoning.

multi-turn model generates responses sequentially with unary feedback after each attempt. In both cases, success is recorded if any of the 5 responses is correct. As shown in Figure 3, multi-turn training consistently outperforms single-turn RL, **achieving up to 14% higher success rate** with similar inference cost. This highlights the benefit of iterative revision under sparse feedback.

Furthermore, we conducted additional experiments comparing various multi-turn training budgets ($T_{\text{max}} = 1, 5, 10$) while consistently using a 5-turn validation setup. Findings presented in Figure 4 demonstrate that larger training budgets yield enhanced performance relative to the single-turn baseline. Notably, both the $T_{\text{max}} = 10$ and $T_{\text{max}} = 5$ configurations delivered **more than a 6% relative improvement** over single-turn training at their peak, clearly emphasizing the benefits of multi-turn training.

To further validate the robustness of multi-turn training benefits, we expanded our analysis by evaluating peak-performing models trained with $T_{\text{max}} \in 1, 5, 10$ across varied inference-time interaction budgets ($k \in 1, 2, 4, 6, 8, 10$). Results illustrated in Figure 5 reinforce previous observations, consistently showing superior Succ@k performance by models trained under multi-turn conditions.Intriguingly, these improvements are observable even at the lowest inference budget ($k = 1$), suggesting that multi-turn training enhances not only iterative performance but also generalizes well to single-shot scenarios. These findings suggest that multi-turn training cultivates robust and flexible reasoning capabilities adaptable to varying conversational interaction depths.

4.2.2 Multi-turn Setting Enables LRM to Revise From Feedback

The multi-turn setting enables agents to engage repeatedly with each prompt (up to T_{max} turns), allowing for richer, more informative interaction trajectories from the same training data. This enhanced utilization of feedback is hypothesized to extract more meaningful learning signals per problem, potentially improving solution quality and accelerating convergence, especially in data-limited contexts.

To empirically validate that LRMs can be improved effectively utilizing conversational feedback for revision, we compared 5-turn training scenarios with and without explicit feedback prompts. Results presented in Figure 6(a) confirm our hypothesis, **demonstrating over 8% peak performance improvement when explicit feedback is provided.**

An additional analysis with feedback prompt only in training (Figure 6(b)) revealed performance improvement as well. This suggests that multi-turn training can even intrinsically enhance model reasoning capabilities.

Finally, our robustness analyses in the Figure 7 indicate that the effectiveness of our method is maintained across various prompt formulations, underscoring its practical applicability in real-world scenarios.

4.2.3 Reward Decay Encourages Efficient Problem Solving

We investigate how different reward schedules influence the agent's learning behavior, particularly in encouraging early success versus allowing extended exploration. All schedules define a reward $r(n)$ based on the turn index n when the first correct answer is produced, with $n \in \{1, \ldots, T_{\text{max}}\}$.

We define and evaluate three distinct reward schedules:

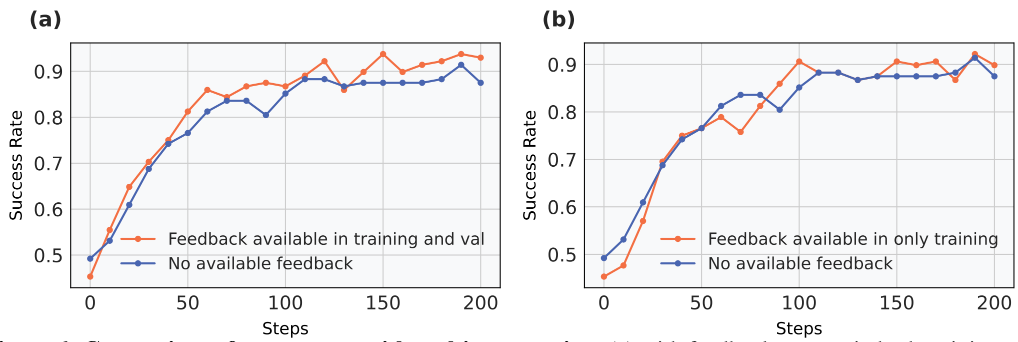

Figure 6: Comparison of success rate with multi-turn setting. (a) with feedback prompt in both training and validation compared to blank prompt; (b) with feedback prompt only in training compared to blank prompt.

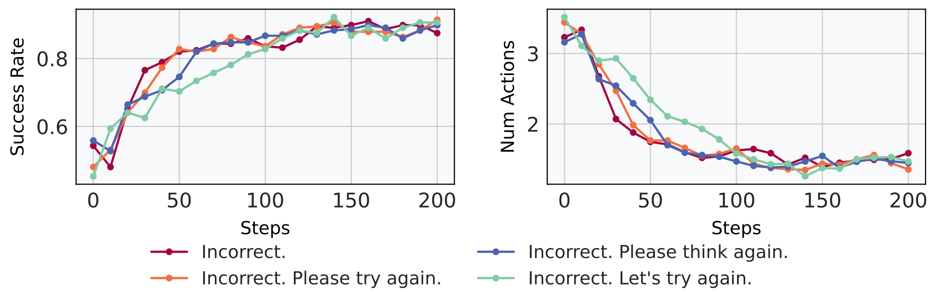

Figure 7: Evaluation under different verbal feedback prompts. Success rates and action counts remain consistent across all variants, demonstrating UFO's robustness to various prompts.

- **Exponential Decay (r_{exp}):** $r_{\text{exp}}(n) = 2^{-(n-1)}$. This schedule imposes a strong penalty for delayed success, with the reward halving for each additional turn taken. It maximally incentivizes the agent to find the solution in the earliest possible turn.
- **Linear Decay (r_{lin}):** $r_{\text{lin}}(n) = \max(0, 1 - 0.2(n-1))$. This provides a gentler, linear decrease in reward for each turn up to 5 turns, after which the reward is zero. This offers a balanced incentive for early solutions without excessively penalizing slightly later successes compared to the exponential schedule.
- **Constant Reward (r_{const}):** $r_{\text{const}}(n) = 1$. This schedule assigns a constant reward for success regardless of the turn number (up to T_{max}). It serves as a baseline to evaluate the impact of the decaying schedules, reflecting a scenario where only task completion matters, not speed.

All schedules operate for $n \in 1, \ldots, T_{\text{max}}$. The agent's objective remains to maximize the expected cumulative reward.

Experimental validation (Figure 8) confirms that **exponential reward decay notably reduces the mean number of actions by roughly 10%**, without sacrificing overall success rates. This reduction in action count suggests that the exponential decay schedule encourages the model to engage in more profound self-reflection and systematic thinking before generating a response. By compelling the model to find solutions in fewer turns, it learns to be more deliberate and efficient, thus minimizing redundant interactions.

By considering the normalized penalty term in our experiment (Equation 8), we count the number of non-repetitive answer for each validation round, as shown in Figure 9. We can tell the percentage increases from 80% to 90%, suggesting that the model performs better in the later stages of training as the model learned to generate different responses better, reducing duplicate answers. This is an important measure of model performance, as high repetition rates lead to higher penalties and thus lower overall rewards. The chart show that the model did improve in this area during training.

5 Related Work

Reinforcement Learning for Enhancing LLM Reasoning and Multi-Turn Interactions. Reinforcement learning from human feedback (RLHF) established that language models can be aligned with user preferences through iterative fine-tuning in natural-language tasks [21–24]. Follow-up work verified that RLHF improves instruction following and already produces rudimentary

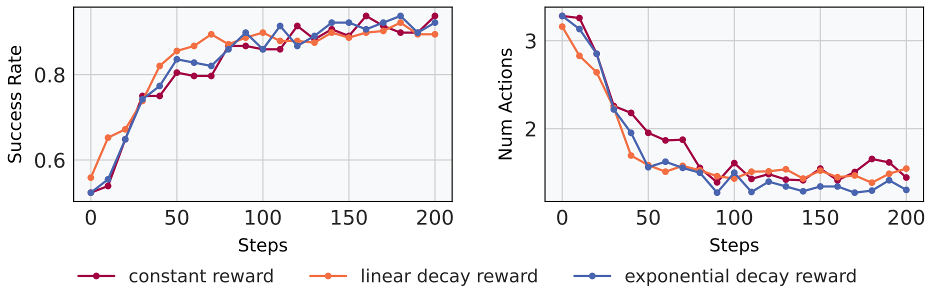

Figure 8: Comparison of reward shaping strategies. While constant, linear decay, and exponential decay schedules achieve similar success rates (left), exponential decay consistently leads to fewer actions per episode (right), indicating more efficient problem solving with less external supervision.

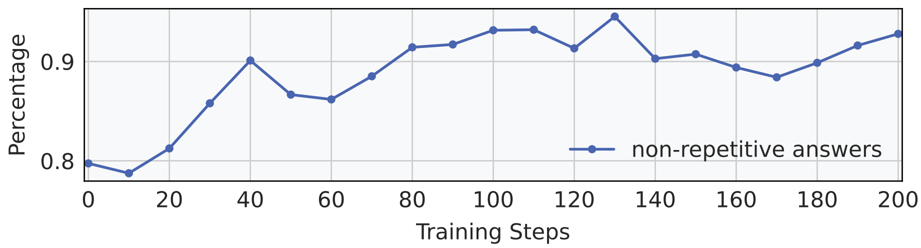

Figure 9: Proportion of non-repetitive answers over training. The upward trend suggests improved diversity across turns, which reduces penalty from repeated responses and contributes to higher overall rewards.

multi-step explanations in summarization and narrative settings [25, 26]. To lower annotation cost, *Reinforcement Learning from AI Feedback* (RLAIF) replaces human labels with synthetic preferences [27], while scalable-oversight *debate* protocols investigate alignment under weak judges [28]. At the optimisation level, lightweight objectives such as Direct Preference Optimisation (DPO) [29, 30], Parameter-Efficient RLHF [31] and Self-Play Fine-Tuning (SPIN) [32] eliminate most on-policy roll-outs. Memory-augmented agents—POEM [33], Larimar [34] and Echo [35]—store episodic context, while self-feedback frameworks such as Self-Refine [36] and CRITIC [37] iteratively repair their own outputs. Real-world applications remain inherently interactive: they expose models to stateful environments, long-horizon credit assignment and delayed rewards [38–40]. Frameworks like Reflexion [41], hierarchical ArCHer [42], search-based Graph-of-Thought [43] and MCTS Self-Refine [44], together with self-play benchmarks such as UNO Arena [45], push multi-turn RL towards robust dialogue, planning and embodied control.

Advancements in Mathematical Reasoning with Large Language Models. LLMs have advanced rapidly on mathematical benchmarks, from grade-school GSM8K [46] to Olympiad-level MATH [47]. Prompting innovations, like Chain-of-Thought [48], its self-consistent variant [49], and Tree/Graph-of-Thought [50, 43], made intermediate reasoning explicit. ReAct interleaves reasoning with environment actions [51], while tool-coupled approaches such as PAL [52], PoT [53] and Toolformer [54] off-load heavy computation. Verifier pipelines boost reliability: from Let's Verify Step-by-Step [55] to AutoPSV [56], MATH-Shepherd [57] and progress-aware verifiers [58]. Binary-search debuggers like URSA locate first-error steps [59]. Nevertheless, intrinsic self-correction remains limited [60]. On the scale axis, domain-specialised pre-training (Minerva [61]) and massive formal corpora (LeanNavigator [62]) provide richer data.

6 Conclusions and Limitations

In this work, we highlight a critical limitation of current single-turn RL training: its tendency to impair multi-turn reasoning by promoting repetitive and shallow responses. To address this, we propose Unary Feedback as Observation (UFO), a simple yet effective method that integrates minimal feedback into existing RL pipelines. UFO enables models to recover and improve both single-turn and multi-turn reasoning performance. Our experiments show a 14% gain in multi-turn accuracy while preserving single-turn quality. Additionally, we demonstrate that incorporating reward decay and repetitive penalty encourages deeper reasoning, self-correction and generating different responses. Our approach is lightweight, generalizable, and easily applicable to existing datasets. A limitation of our work is its primary focus on mathematical reasoning tasks, leaving its generalizability to broader reasoning domains for future investigation.

References

[1] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948. 1, 2.1

[2] OpenAI. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

[3] Qwen, :, 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, Tianyi Tang, 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, 2025. URL https://arxiv.org/abs/2412.15115.

[4] Gemini Team. Gemini: A family of highly capable multimodal models, 2025. URL https://arxiv.org/abs/2312.11805. 1

[5] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024. URL https://arxiv.org/abs/2404.07972. 1, 1, 2.1

[6] Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with swe-gym, 2024. URL https://arxiv.org/abs/2412.21139. 1, 2.1

[7] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents, 2023. URL https://arxiv.org/abs/2207.01206.

[8] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning, 2021. URL https://arxiv.org/abs/2010.03768. 1

[9] Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. Mint: Evaluating llms in multi-turn interaction with tools and language feedback, 2024. URL https://arxiv.org/abs/2309.10691. 1, 1, 2.1

[10] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347. 1, 2.1

[11] Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl, 2024. URL https://arxiv.org/abs/2402.19446. 1

[12] Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Monica Lam, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning, 2025. URL https://arxiv.org/abs/2504.20073. 1, 2.1, 3.3

[13] Yan Zhuang*, Jiawei Ren*, Xiaokang Ye*, Xuhong He, Zijun Gao, Ryan Wu, Mrinaal Dogra, Cassie Zhang, Kai Kim, Bertt Wolfinger, Ziqiao Ma, Tianmin Shu†, Zhiting Hu†, and Lianhui Qin†. Simworld: A world simulator for scaling photorealistic multi-agent interactions, 2025. 1

[14] Shiyi Cao, Sumanth Hegde, Dacheng Li, Tyler Griggs, Shu Liu, Eric Tang, Jiayi Pan, Xingyao Wang, Akshay Malik, Graham Neubig, Kourosh Hakhamaneshi, Richard Liaw, Philipp Moritz, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Skyrl-v0: Train real-world long-horizon agents via reinforcement learning, 2025. 1

[15] Qineng Wang Pingyue Zhang Linjie Li Zhengyuan Yang Kefan Yu Minh Nhat Nguyen Licheng Liu Eli Gottlieb Monica Lam Yiping Lu Kyunghyun Cho Jiajun Wu Li Fei-Fei Lijuan Wang Yejin Choi Manling Li Zihan Wang, Kangrui Wang. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning, 2025. URL https://arxiv.org/abs/2504.20073. 1, 4.1

[16] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155. 2.1

[17] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300. 2.1

[18] 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, 2025. URL https://arxiv.org/abs/2503.09516. 2.1

[19] Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. Retool: Reinforcement learning for strategic tool use in llms, 2025. URL https://arxiv.org/abs/2504.11536. 2.1

[20] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model, 2025. URL https://arxiv.org/abs/2503.24290. 3.3

[21] Jeffrey Wu Tom B. Brown Alec Radford Dario Amodei Paul Christiano Geoffrey Irving Daniel M. Ziegler, Nisan Stiennon. Fine-tuning language models from human preferences, 2019. URL https://arxiv.org/abs/1909.08593. 5

[22] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

[23] Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2017. URL https://arxiv.org/abs/1706.03741.

[24] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners, 2023. URL https://arxiv.org/abs/2205.11916. 5

[25] Jeff Wu Daniel M. Ziegler Ryan Lowe Chelsea Voss Alec Radford Dario Amodei Paul Christiano Nisan Stiennon, Long Ouyang. Learning to summarize from human feedback, 2020. URL https://arxiv.org/abs/2310.03708. 5

[26] Daniel M. Ziegler Nisan Stiennon Ryan Lowe Jan Leike Paul Christiano Jeff Wu, Long Ouyang. Recursively summarizing books with human feedback, 2021. URL https://arxiv.org/abs/2109.10862. 5

[27] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, et al. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback, 2023. URL https://arxiv.org/abs/2309.00267. 5

[28] Anton Bakhtin, Jeffrey Ling, Jack Hessel, Ari Holtzman, Sam McCandlish, et al. On scalable oversight with weak llms judging strong llms, 2024. URL https://arxiv.org/abs/2407.04622. 5

[29] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023. URL https://arxiv.org/abs/2305.18290. 5

[30] Jing Shao Xiangyu Yue Chao Yang Wanli Ouyang Yu Qiao Zhanhui Zhou, Jie Liu. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization, 2023. URL https://arxiv.org/abs/2310.03708. 5

[31] Yihe Deng, Yujia Xu, Shangqian Chen, and Kai-Wei Chang. Parameter efficient reinforcement learning from human feedback, 2024. URL https://arxiv.org/abs/2403.10704. 5

[32] Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models, 2024. URL https://arxiv.org/abs/2401.01335. 5

[33] Hieu D. Do, Chunting Zhou, and Pengfei Liu. Large language models prompting with episodic memory, 2024. URL https://arxiv.org/abs/2408.07465. 5

[34] Lingpeng Liu, Sam Roberts, Yuhuai Wu, et al. Larimar: Large language models with episodic memory control, 2024. URL https://arxiv.org/abs/2403.11901. 5

[35] Haoyang Zhou, Qianjiang Wang, Yuwei Wang, et al. Echo: A large language model with temporal episodic memory, 2025. URL https://arxiv.org/abs/2502.16090. 5

[36] Aman Madaan, Uri Alon, Yiming Yang, Graham Neubig, and Zico Kolter. Self-refine: Iterative refinement with self-feedback, 2023. URL https://arxiv.org/abs/2303.17651. 5

[37] Linyi Gou, Yang Song, Kun Zhou, et al. Critic: Large language models can self-correct with tool-augmented critiquing, 2023. URL https://arxiv.org/abs/2305.11738. 5

[38] Richard S Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018. 5

[39] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

[40] Yun Qu, Yuhang Jiang, Boyuan Wang, Yixiu Mao, Cheems Wang, et al. Latent reward: Llm-empowered credit assignment in episodic reinforcement learning, 2024. URL https://arxiv.org/abs/2412.11120. 5

[41] Edward Berman Ashwin Gopinath Karthik Narasimhan Shunyu Yao Noah Shinn, Federico Cassano. Reflexion: Language agents with verbal reinforcement learning, 2023. URL https://arxiv.org/abs/2303.11366. 5

[42] Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl, 2024. URL https://arxiv.org/abs/2402.19446. 5

[43] Ronald Peng, Zijun Xue, Yujia Qin, and Chen Liang. Graph of thoughts: Solving elaborate problems with large language models, 2023. URL https://arxiv.org/abs/2308.09687. 5

[44] Long Ouyang, B. Li, Prescott T. Ulmer, et al. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree search self-refinement, 2024. URL https://arxiv.org/abs/2406.07394. 5

[45] Zhanyue Qin, Haochuan Wang, Deyuan Liu, et al. Uno arena for evaluating sequential decision-making capability of large language models, 2024. URL https://arxiv.org/abs/2406.16382. 5

[46] Karl Cobbe, Jacob Hilton, Reiichiro Nakano, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168. 5

[47] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, et al. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874. 5

[48] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, et al. Chain-of-thought prompting elicits reasoning in large language models, 2022. URL https://arxiv.org/abs/2201.11903. 5

[49] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain-of-thought reasoning in language models, 2022. URL https://arxiv.org/abs/2203.11171. 5

[50] Shunyu Yao, Dian Yu, Jeffrey Zhang, Yuan Yang, Daniel Khashabi, et al. Tree of thoughts: Deliberate problem solving with large language models, 2023. URL https://arxiv.org/abs/2305.10601. 5

[51] Dian Yu Nan Du Izhak Shafran Karthik Narasimhan Yuan Cao Shunyu Yao, Jeffrey Zhao. React: Synergizing reasoning and acting in language models, 2022. URL https://arxiv.org/abs/2210.03629. 5

[52] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, et al. Pal: Program-aided language models, 2022. URL https://arxiv.org/abs/2211.10435. 5

[53] Zhuohan Chen, Kaili Huang, Teddy Lee, Zhen Zhang, Lindsay Lamont, et al. Program of thoughts prompting: Disentangling computation from reasoning, 2023. URL https://arxiv.org/abs/2211.12588. 5

[54] Timo Schick, Jane Dwivedi-Yu, Ruiqi Zhong, Dinghan Shen, Hin Richter, et al. Toolformer: Language models can teach themselves to use tools, 2023. URL https://arxiv.org/abs/2302.04761. 5

[55] Sam Lightman, John Schulman, Jacob Hilton, et al. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050. 5

[56] Qijing Li, Yuchen Li, and Minlie Huang. Autopsv: Automated process-supervised verifier, 2024. URL https://arxiv.org/abs/2405.16802. 5

[57] Shuchen Wang, Linyi Yang, Linfeng Zhang, et al. Math-shepherd: Verify and reinforce llms step by step, 2024. URL https://arxiv.org/abs/2312.08935. 5

[58] Cheng Zhang, Detian Deng, and Tao Lei. Rewarding progress: Scaling automated process verifiers for llm reasoning, 2024. URL https://arxiv.org/abs/2410.08146. 5

[59] Jiacheng Liu, Haoran Li, Tianyu Du, et al. Ursa: Understanding and verifying chain-of-thought reasoning in large language models, 2024. URL https://arxiv.org/abs/2501.04686. 5

[60] Junnan Huang, Zhihao Zhou, Zihang Long, et al. Large language models cannot self-correct reasoning yet, 2023. URL https://arxiv.org/abs/2310.01798. 5

[61] Minjia Huang, Noah Friedman, Ethan Dyer, S. Dhebar, Daniel Gilat, et al. Solving quantitative reasoning problems with language models, 2022. URL https://arxiv.org/abs/2206.14858. 5

[62] Kaiyang Yang, Yuxuan Jiang, Jialin Song, et al. Generating millions of lean theorems with proofs by exploring libraries, 2025. URL https://arxiv.org/abs/2503.04772. 5

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state our problem scope and contributions.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to Section 6.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper is primarily empirical and does not present new theoretical results with formal proofs.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please refer to Section 4.1.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We use publicly-accessable environments, as detailed in Section 4.1. We upload the codes and instructions to recover the results. Once the blind review period is finished, we'll open-source all codes and instructions.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please refer to Section 4.1.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: LLM training and inference tasks are very resource intensive and costly.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to Section 4.1.

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: We followed the NeurIPS Code of Ethics.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: Our work focuses on academic, publicly-available evaluation suites and environments. This work is not related to any private or personal data, and there's no explicit negative social impacts.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: We do not foresee any high risk for misuse of this work.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We credit the existing assets in appropriate ways.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper primarily introduces methods and evaluations on existing or simulated environments, not new standalone assets for release.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The research described does not involve crowdsourcing or direct human subject experimentation.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The research described does not involve human subjects, so IRB approval was not applicable.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: The core of the research involves the study Large Language Models. Their use is central to the methodology.