T^3: Reducing Belief Deviation in Reinforcement Learning for Active Reasoning

Deyu Zou[1]†, Yongqiang Chen[1]†, Jianxiang Wang[2], Haochen Yang[1], Mufei Li[3], Qing Da[2], Pan Li[3], Yu Gong[2], James Cheng[1]
[1]The Chinese University of Hong Kong, [2]ByteDance, [3]Georgia Institute of Technology
`dyzou24@cse.cuhk.edu.hk`, `yuxiaofei@bytedance.com`
† Equal Contribution.

Abstract

Active reasoning requires Large language models (LLMs) to interact with external sources and gather missing information to solve a problem. Reinforcement learning with outcome reward, as a *de facto* approach to incentivize active reasoning of LLMs, however, often loses track of problem states and generates uninformative and repetitive actions. Consequently, it leads to *more and more belief deviation* – the divergence between the oracle belief and the agent's internal belief state. To mitigate the issue, it is essential to properly assign rewards to and promote intermediate steps that are more purposeful and informative in solving the problem while avoiding being trapped by cumulative belief deviation. As directly tracking the deviation of belief states is intractable, we introduce $\mathbf{T}^3$, which leverages proxy signals of excessive belief deviation to assign intermediate rewards or directly truncates the rollout trajectories during training. Across two recent datasets tailored for active reasoning, $\mathbf{T}^3$ improves both performance and stability of diverse RL algorithms, achieving gains up to 30%. These results highlight belief control as a key principle for training robust LLM-based active reasoners.

1 Introduction

Large language models (LLMs) have demonstrated remarkable reasoning capabilities across a wide range of domains [8, 11, 15], where reinforcement learning (RL) with outcome rewards further pushes forward the frontier of reasoning models [6, 14, 20, 21, 25, 27]. Yet, the existing paradigm mostly focuses on *passive reasoning*, where the LLM agent is given all the necessary information to derive the correct conclusion in *single-turn interaction*. However, many real-world applications are different, where the agent has to *actively acquire* the desired knowledge through *multi-turn interactions* with external sources. It motivates a new paradigm of reasoning, i.e., *active reasoning* [7, 13, 34].

Active reasoning requires the agent to proactively identify the *missing information* for reasoning and propose *purposeful actions*, such as targeted queries, clarification requests, or evidence gathering, etc., to reduce uncertainty and advance toward the solution. RL with outcome rewards appears to be a natural framework to incentivize active reasoning capabilities, as in passive reasoning. Yet, LLMs often struggle in maintaining a reliable internal representation of the problem state, and outcome-based rewards can hardly assign credits to key steps [4, 9, 23]. Consequently, it introduces unstable learning dynamics and hinders policy convergence, and leads to more and more *belief deviation* – the divergence between the oracle belief and the agent's internal belief state. When with excessive belief deviation, the agent tends to generate actions that are redundant, irrelevant, or uninformative [5, 31, 32] or even collapse into unproductive cycles [34]. The above-mentioned challenges raise a critical research question: *Which design would prevent unhelpful actions from contaminating the credit assignment, thereby enabling stable and effective RL for active reasoning?*

To answer the question, we first show that active reasoning can be naturally formulated as a Partially Observable Markov Decision Process (POMDP), where the central difficulty arises from the belief modeling. Once there is a large belief deviation, the agent is prone to issuing uninformative actions, triggering error accumulation, and leading the trajectory into non-viable states, unable to reach the final solution. Therefore, we develop $\mathbf{T^3}$ (Truncating Belief-Trapped Trajectories), a simple yet effective early-truncation mechanism that detects and leverages *proxy signals* when the model belief deviates substantially from the oracle belief. Although precisely quantifying beliefs in LLM-based reasoning is intractable, we find that in every concrete setting, the model exhibits recognizable and reliable patterns of belief deviation. When the intermediate steps comply with little belief deviation and facilitate the solving of the problem, $\mathbf{T^3}$ assigns credits and promotes the exploitation of those steps. When there exists a large belief deviation, $\mathbf{T^3}$ directly halts the trajectory before the reasoning process gets trapped and error accumulation overwhelms credit assignment.

We evaluate $\mathbf{T^3}$ on two challenging datasets from AR-BENCH [34], GUESSNUMBERS, and SITUATIONPUZZLES, evaluated on a wide range of recent RL algorithms. Across all settings, $\mathbf{T^3}$ consistently improves both training stability and final performance, achieving gains of up to 30%. These results highlight that controlling belief deviation not only alleviates the credit assignment pathology but also offers a promising direction for building reliable active reasoning agents.

2 Methodology

Preliminaries and Problem Formulations. We model the problem of active reasoning as a Partially Observable Markov Decision Process (POMDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, T, O, R, \gamma)$, where $\mathcal{S}$ is the space of unobservable latent states, $\mathcal{A}$ the action space, $\mathcal{O}$ the observation space, $T(s' \mid s, a)$ the transition dynamics, $O(o \mid s, a)$ the observation model, R the reward function, and γ the discount factor. At each step t, an ideal Bayesian reasoner would maintain a *oracle belief* distribution $b_t^* \in \Delta(\mathcal{S})$, *i.e.*, a posterior over latent states given the full history of actions and observations. It is recursively updated via the Bayesian paradigm:

$$b_{t+1}^*(s') = \eta \cdot O(o_t \mid s', a_t) \sum_{s \in \mathcal{S}} T(s' \mid s, a_t) b_t^*(s), \tag{1}$$

where η is a normalization constant. In contrast, an LLM policy π_θ does not implement the precise Bayesian filtering. Instead, it maintains the *agent belief state* b_t, which encodes the agent's internal representation of what the latent state could be and what information remains missing. Concretely, the induced trajectory distribution is $\tau = (b_1, a_1, o_1, b_2, a_2, o_2, \ldots, b_T, a_T, o_T)$ where b_t can be implicitly maintained inside the LLM or explicitly in the trajectory as the format such as Chain-of-Thought [26]. The generative process is thus given by

$$\Pr(\tau; \pi_\theta) = \prod_{t=1}^{T} \underbrace{\pi_\theta(b_t \mid b_{t-1}, a_{t-1}, o_{t-1})}_{\text{agent belief transition}} \cdot \underbrace{\pi_\theta(a_t \mid b_t)}_{\text{action generation}} \cdot O(o_t \mid s_t, a_t). \tag{2}$$

2.1 Belief Deviation and its Challenges

We define the *belief deviation* at step t as the divergence between the oracle belief and the agent-induced belief: $d_t = D\big(b_t \,\|\, b_t^*\big)$, where $D(\cdot|\cdot)$ is a divergence measure (*e.g.*, the KL divergence). Let $\pi^*(\cdot \mid b_t^*)$ denote the optimal policy conditioned on the oracle belief, and $\pi(\cdot \mid b_t)$ the agent's policy under its internal belief. Due to the policy's sensitivity, the policy mismatch $\Delta_t := \mathrm{TV}\left(\pi\left(\cdot \mid b_t\right) \| \pi^*\left(\cdot \mid b_t^*\right)\right)$ grows with the belief deviation d_t, where $\mathrm{TV}(\cdot\|\cdot)$ denotes the total variation. This distortion in turn increases the likelihood of selecting uninformative actions. Crucially, such actions do not update the belief state in a meaningful way, yielding $b_{t+1} \approx b_t$, while the oracle state belief b_{t+1}^* evolves, further amplifying the belief deviation. This creates a vicious self-reinforcing cycle of error accumulation. Beyond a critical threshold, the trajectory enters a "trapped" regime in which progress towards task completion becomes impossible.

Challenges – Credit Assignment Issue. This phenomenon complicates *credit assignment*. Consider a rollout of length T, where belief deviation becomes critical at step $t^\star$. From that point onward,

Table 1: Experimental results of RL algorithms with and without $\mathbf{T^3}$ on two active reasoning datasets.

	SITUATIONPUZZLES			GUESSNUMBERS	
	F1-word	F1-char	Turn Counts	Exact Match	Turn Counts
Inferene (Zero-shot)	19.46	41.62	6.53	20.94	3.98
PPO	28.77	74.56	5.79	91.62	6.19
PPO *w.* $\mathbf{T^3}$	36.85	81.5	3.1	93.98	3.65
GRPO	36.46	83.73	7.89	61.26	3.38
GRPO *w.* $\mathbf{T^3}$	39.45	84.58	11.41	91.36	3.88
GSPO	36.63	82.17	3.17	96.07	4.05
GSPO *w.* $\mathbf{T^3}$	36.96	82.08	2.3	99.74	3.71

actions $\{t^\star, \ldots, T\}$ contribute little to task progress, yet the final reward R_T is still attributed to every step. As a result, the genuinely informative early actions ($< t^\star$) receive diluted or even misleading credit. This misattribution hinders learning, since the policy is updated with gradients contaminated by uninformative late actions. See the detailed and exemplified analyses in Appendix C.1.

2.2 $\mathbf{T^3}$: Truncating Belief-Trapped Trajectories

To mitigate the challenges caused by belief deviation, we propose $\mathbf{T^3}$, an early-truncation mechanism for RL-based active reasoning. The key idea is to terminate rollouts once belief deviation d_t becomes excessively large, thereby preventing the trapped belief and uninformative steps from contaminating credit assignment. Formally, given a threshold $\delta > 0$, we define a stopping time $t_{\text{stop}} = \min\{t : d_t > \delta\}$, and truncate the trajectory at t_{stop}, assigning the final reward $R(\tau_{\text{stop}})$ accordingly. This effectively converts states with excessive deviation into *absorbing terminals*, ensuring that downstream learning signals remain focused on informative reasoning segments.

Practical Considerations. In practice, the latent state space $\mathcal{S}$ can be extremely large and semantically complex, making it intractable for exact Bayesian updates and a direct computation of b_t^* and d_t. Instead, we exploit the empirical observation that LLMs exhibit *recognizable patterns of belief deviation* in concrete reasoning settings (*c.f.*, Sec 3.2). These observable patterns serve as *proxy conditions* for belief deviation and trigger early truncation. During RL training, rollout generation halts whenever a proxy condition is detected, and the final reward is back-propagated accordingly.

3 Experiments

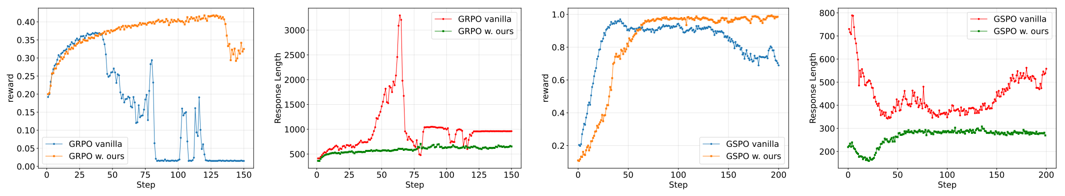
3.1 Experimental Setup

Datasets. We evaluate $\mathbf{T^3}$ on two challenging datasets from AR-Bench [34], a recent benchmark on evaluating LLMs' active reasoning capabilities: (I) Based on its GUESSNUMBERS (GN) dataset, we rebuild and construct the $\text{GN}(a, b)$ series, where the goal is to deduce a hidden a-digit number composed of b unique symbols. The LLM solver interacts by making guesses and receives structured feedback about the gap between the current guess and the answer. (II) We directly use its SITUATIONPUZZLES (SP) dataset, where an LLM solver must unravel a paradoxical puzzling scenario through iterative yes-no questioning and feedback gathering. See more details in Appendix B.1.

Baselines. To evaluate the effectiveness of $\mathbf{T^3}$, we compare it against the following baselines: 1) Direct Inference without Training, 2) PPO [17], 3) GRPO [18], and 4) GSPO [33]. PPO and GRPO are widely adopted RL methods for enhancing the reasoning capabilities of LLMs. GSPO is a recently proposed method by the Qwen team that has drawn attention. See more details in Appendix B.2.

Implementation Details. The main experiments of RL training are conducted on Qwen2.5-7B-Instruct [28]. For the GN dataset, the interactive feedback is rule-based; for the SP dataset, a Qwen2.5-14B-Instruct model provides the interactive feedback. The maximum allowed interaction turn counts of GN and SP are 10 and 15, respectively. See more implementation details in Appendix B.3.

Evaluation Metrics. For the GN dataset, we utilize *Exact Match*, measuring whether the final prediction made by the LLM exactly matches the target number; For the SP dataset, we report the *F1* score (including word-level and character-level) to assess similarity between the ground truth and the solution made by the LLM. In addition, for both datasets, we utilize *Turn Counts*, which count the total rounds of LLM-environment interaction until the game is successfully solved.



(a) Reward (SP) (b) Response Length (SP) (c) Reward (GN) (d) Response Length (GN)

Figure 1: Dynamics of reward and response length in the SP (a/b) on GRPO and GN (c/d) on GSPO.

3.2 Algorithm Specification

For the main experiments reported in Sec. 3.3, we implement $\mathbf{T^3}$ with task-specific proxy conditions for belief deviation on two evaluated datasets as follows. We are aware that there exist alternative choices and the ablations of different proxies are shown in Appendix D.1.

SP. A trajectory is truncated if the judge model returns "no" or "unknown" as feedback for five consecutive turns. This proxy effectively detects a persistent lack of progress, signaling that the agent's belief has likely deviated and is trapped in an unproductive line of questioning.

GN. The task involves deducing a hidden number, and the solution space can be iteratively filtered based on the feedback from each guess. Let S_t denote the feasible solution space at turn t, which is initialized to all possible numbers. After each guess a_t and its feedback observation o_t, the space is updated to be consistent with the new information: $S_t := \{g \in S_{t-1} \mid \texttt{Feedback}(g, a_t) \sim o_t\}$. We define a belief deviation if the agent's guess a_t falls outside the current feasible set S_{t-1}. This condition enforces that the agent's guesses remain logically consistent with the gathered information.

3.3 Experimental Results and Analyses

Overall Performance. The main experimental results can be seen in Table 1. All RL-trained agents, both with and without $\mathbf{T^3}$, drastically outperform the zero-shot inference baseline, underscoring the necessity of RL in incentivizing LLMs' active reasoning capabilities. Integrating $\mathbf{T^3}$ consistently boosts the final task performance of all evaluated RL algorithms. Notably, on the GN dataset, $\mathbf{T^3}$ helps GSPO achieve a near-perfect 99.74% Exact Match accuracy. A substantial improvement is also observed with GRPO, where $\mathbf{T^3}$ leads to a remarkable 30.1% gain. On the SP dataset, $\mathbf{T^3}$ yields the best overall performance on F1-word and F1-char scores with GRPO.

Better Stability and Optimality of Training. Beyond final performance, $\mathbf{T^3}$ substantially improves training dynamics. As shown in Fig. 1a and 1c, vanilla RL methods for active reasoning exhibit higher variance and instability, with rewards often collapsing after partial convergence. By contrast, $\mathbf{T^3}$ enables them to maintain monotonic or near-monotonic reward improvement without catastrophic drops (or at much later steps). Therefore, agents not only converge more reliably but also reach higher optima. These results highlight the dual benefit of $\mathbf{T^3}$: stabilizing reinforcement learning while guiding policies toward more informative and effective reasoning behaviors.

Higher Token Efficiency of Training. While the reward dynamics *wrt.* step (Fig. 1a and 1c) seem to indicate that RL with $\mathbf{T^3}$ achieves slightly slower reward growth, early truncation ensures that each rollout consumes fewer tokens on average (*c.f.*, Fig. 1b and 1d), and therefore, our method actually exhibits higher token efficiency overall. Take the cases shown in Fig. 1 as an example. Under GRPO on SP, to reach a reward level of 0.36, our method consumes 96.6% of the total tokens compared to vanilla on average; under GSPO on GN, to reach 0.96, it requires merely 76.3% of the tokens. More importantly, while vanilla methods stagnate and fail to improve further, our method continues to enhance rewards, achieving up to 0.41 on SP and 0.99 on GN.

4 Conclusion

In this work, we identified belief deviation as a key cause of training instability and suboptimality in RL for active reasoning of LLMs. To mitigate its harmful accumulation, we introduced $\mathbf{T^3}$, a simple yet effective early-truncation mechanism that halts the trapped trajectories. Empirical results on AR-BENCH show that $\mathbf{T^3}$ consistently enhances the stability and performance of diverse RL algorithms. This work establishes belief deviation as a critical failure mode and demonstrates that its control is a powerful principle for cultivating robust active reasoning in LLMs.

References

[1] Kartikeya Badola, Jonathan Simon, Arian Hosseini, Sara Marie Mc Carthy, Tsendsuren Munkhdalai, Abhimanyu Goyal, Tomáš Kočiskỳ, Shyam Upadhyay, Bahare Fatemi, and Mehran Kazemi. Multi-turn puzzles: Evaluating interactive reasoning and strategic dialogue in llms. *arXiv preprint arXiv:2508.10142*, 2025.

[2] Yang Deng, Lizi Liao, Liang Chen, Hongru Wang, Wenqiang Lei, and Tat-Seng Chua. Prompting and evaluating large language models for proactive dialogues: Clarification, target-guided, and non-collaboration. *arXiv preprint arXiv:2305.13626*, 2023.

[3] Shihan Dou, Yan Liu, Haoxiang Jia, Limao Xiong, Enyu Zhou, Wei Shen, Junjie Shan, Caishuang Huang, Xiao Wang, Xiaoran Fan, et al. Stepcoder: Improve code generation with reinforcement learning from compiler feedback. *arXiv preprint arXiv:2402.01391*, 2024.

[4] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*, 2025.

[5] Dayuan Fu, Keqing He, Yejie Wang, Wentao Hong, Zhuoma Gongque, Weihao Zeng, Wei Wang, Jingang Wang, Xunliang Cai, and Weiran Xu. Agentrefine: Enhancing agent generalization through refinement tuning. *arXiv preprint arXiv:2501.01702*, 2025.

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

[7] Zhiyuan Hu, Chumin Liu, Xidong Feng, Yilun Zhao, See-Kiong Ng, Anh Tuan Luu, Junxian He, Pang Wei W Koh, and Bryan Hooi. Uncertainty of thoughts: Uncertainty-aware planning enhances information seeking in llms. *Advances in Neural Information Processing Systems*, 37:24181–24215, 2024.

[8] Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

[9] Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Unlocking rl potential for llm reasoning through refined credit assignment. 2024.

[10] Wai-Chung Kwan, Xingshan Zeng, Yuxin Jiang, Yufei Wang, Liangyou Li, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. Mt-eval: A multi-turn capabilities evaluation benchmark for large language models. *arXiv preprint arXiv:2401.16745*, 2024.

[11] Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025.

[12] Zhenwen Liang, Dian Yu, Wenhao Yu, Wenlin Yao, Zhihan Zhang, Xiangliang Zhang, and Dong Yu. Mathchat: Benchmarking mathematical reasoning and instruction following in multi-turn interactions. *arXiv preprint arXiv:2405.19444*, 2024.

[13] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023.

[14] OpenAI. Openai o3-mini. https://openai.com/index/openai-o3-mini/, January 2025.

[15] Aske Plaat, Annie Wong, Suzan Verberne, Joost Broekens, Niki van Stein, and Thomas Bäck. Reasoning with large language models, a survey. *CoRR*, 2024.

[16] 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.

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

[18] 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.

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

[20] Saksham Sahai Srivastava and Vaneet Aggarwal. A technical survey of reinforcement learning techniques for large language models. *arXiv preprint arXiv:2507.04136*, 2025.

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

[22] Yanming Wan, Jiaxing Wu, Marwa Abdulhai, Lior Shani, and Natasha Jaques. Enhancing personalized multi-turn dialogue with curiosity reward. *arXiv preprint arXiv:2504.03206*, 2025.

[23] Hanlin Wang, Chak Tou Leong, Jiashuo Wang, Jian Wang, and Wenjie Li. Spa-rl: Reinforcing llm agents via stepwise progress attribution. *arXiv preprint arXiv:2505.20732*, 2025.

[24] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

[25] Shuhe Wang, Shengyu Zhang, Jie Zhang, Runyi Hu, Xiaoya Li, Tianwei Zhang, Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy. Reinforcement learning enhanced llms: A survey. *arXiv preprint arXiv:2412.10400*, 2024.

[26] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[27] Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, et al. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*, 2025.

[28] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[29] Haofei Yu, Zhengyang Qi, Yining Zhao, Kolby Nottingham, Keyang Xuan, Bodhisattwa Prasad Majumder, Hao Zhu, Paul Pu Liang, and Jiaxuan You. Sotopia-rl: Reward design for social intelligence. *arXiv preprint arXiv:2508.03905*, 2025.

[30] Yuanqing Yu, Zhefan Wang, Weizhi Ma, Zhicheng Guo, Jingtao Zhan, Shuai Wang, Chuhan Wu, Zhiqiang Guo, and Min Zhang. Steptool: A step-grained reinforcement learning framework for tool learning in llms. 2024.

[31] Siyu Yuan, Zehui Chen, Zhiheng Xi, Junjie Ye, Zhengyin Du, and Jiecao Chen. Agent-r: Training language model agents to reflect via iterative self-training. *arXiv preprint arXiv:2501.11425*, 2025.

[32] Zijing Zhang, Ziyang Chen, Mingxiao Li, Zhaopeng Tu, and Xiaolong Li. Rlvmr: Reinforcement learning with verifiable meta-reasoning rewards for robust long-horizon agents. *arXiv preprint arXiv:2507.22844*, 2025.

[33] Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

[34] Zhanke Zhou, Xiao Feng, Zhaocheng Zhu, Jiangchao Yao, Sanmi Koyejo, and Bo Han. From passive to active reasoning: Can large language models ask the right questions under incomplete information? *arXiv preprint arXiv:2506.08295*, 2025.

A Related Work

Active Reasoning of LLMs. Active reasoning requires LLMs to interact with external sources and actively acquire missing information to solve complex tasks. Prior work has improved LLMs' ability to handle ambiguity and incompleteness through making clarification and information-seeking actions. For example, Proactive CoT [2] prompts LLMs to identify ambiguous problems and generate clarification questions, while UoT [7] quantifies the contribution of each question in reducing uncertainty. However, challenges remain when transitioning from LLMs' single-turn success to multi-turn active reasoning [1, 10, 12], even with several advanced strategies such as tree-based searching or post-training approaches, as highlighted in existing works [34]. In contrast, we leverage reinforcement learning to explicitly incentivize active reasoning capabilities, and propose $\mathbf{T^3}$ to address key issues when applying RL in this setting.

Credit Assignment in Multi-turn RL. Existing methods have extensively explored rule-based approaches [3, 30, 32] to shape intermediate rewards. Several recent works [22, 23, 29] have proposed measuring the progress of stepwise actions toward overall task completion as intermediate rewards. Specifically, CURIO [22] constructs a potential function over an ideal belief state to assign intermediate rewards, assuming that the latent state space is finite and enumerable. Sotopia-RL [29] relies on reward labeling with proprietary LLMs. SPA-RL [23] trains reward models for intermediate rewards by enforcing a summation constraint with respect to the final outcome reward. In our studied active reasoning scenario, belief deviation under partial observability makes it difficult for outcome-based rewards to properly assign credit to key reasoning steps. Our proposed $\mathbf{T^3}$ mitigates this by halting the trajectory before the reasoning process becomes trapped in excessive belief deviation and the error accumulation overwhelms credit assignment.

B Setup Details

B.1 Dataset Details and Prompt Templates

SITUATIONPUZZLES (SP). This task introduces a challenging active reasoning task where the LLM player must uncover a coherent narrative from an initially puzzling scenario. Each puzzle begins with a brief, paradoxical statement. The solver interacts iteratively with a judge by asking binary yes-no questions, gathering feedback from the judge to constrain the solution space. The goal is to formulate a complete and plausible explanation that resolves the apparent contradiction. We directly use this dataset from the AR-Bench [34]. In our experiments, we utilize a Qwen2.5-14B-Instruct model to provide the interactive feedback.

The prompt template for the SITUATIONPUZZLES dataset can be seen in Fig. 3. For SITUATIONPUZZLES, put a specific puzzle to solve into `{puzzle}` of the prompt. The prompt template for the judge LLM is shown in Fig. 5. The judge will receive `{surface}` and `{bottom}` to understand the whole puzzle, and give yes-no feedback according to the player LLM's question.

GUESSNUMBERS (GN). Adapted from the original dataset proposed by AR-Bench [34] which the player must crack a 4-digit secret (digits are unique in 0-9), our newly constructed GN(a, b) is a series of reasoning tasks that involve the LLM agent's interactive deduction with external sources: the target is a a-digit number, where each digit is sampled from a set of b unique symbols without repetition. This yields $P(b, a) = b!/(b - a)!$ possible targets.

At each step, the LLM agent makes a guess and receives structured feedback in the form of `xAyB`, where x denotes the number of digits that are both correct in value and position (denoted as "A"), and y denotes the number of digits that are correct in value but placed in the wrong position (denoted as "B"). The agent is expected to actively perform reasoning based on accumulated observations and interact with an external source to efficiently reduce uncertainty and locate the correct answer.

To control for randomness in the first move, which plays a minor role in evaluating the LLM agent's ability to understand and update based on observations, we fix the first guess to a deterministic number that is guaranteed to differ from the answer. This means we need (a, b, g_0, x_0, y_0) to specify a question for the LLM player, where g_0 denotes the initial guess, and (x_0, y_0) denotes the corresponding initial feedback of the form $\mathtt{x_0Ay_0B}$.

We group data items by their tuple (a, b, x_0, y_0), since items sharing the same (a, b, x_0, y_0) correspond to tasks with similar uncertainty reduction dynamics and reasoning logic patterns. Specifically, our constructed dataset covers all data items of the following sub-groups: $(3,4,0,3)$, $(3,4,2,0)$, $(3,4,1,2)$, $(3,5,1,2)$, $(3,5,0,3)$, $(3,5,1,0)$, $(3,5,2,0)$, $(4,4,0,4)$, and $(4,5,3,0)$. These configurations are carefully selected to ensure diversity in task complexity: varying (a,b) controls the size of the hypothesis space, while varying (x_0,y_0) shapes the initial reasoning landscape by introducing distinct patterns of partial evidence. Finally, we perform a randomized train-test split to the obtained set for training and evaluation.

The prompt template for the GUESSNUMBERS dataset can be seen in Fig. 4. For GUESSNUMBERS, we need to first specify `{num_digits}` and `{num_uniques}`, corresponding to (a,b) mentioned above, and then specify the initial guess in `{initial_guess}`, and the resulting initial feedback in `{initial_feedback_same_pos}` and `{initial_feedback_diff_pos}`.

B.2 Baseline Details

Here we introduce RL algorithms used in our experiments. Formally, given an actor model π_θ, the likelihood of a response y to a query x under the policy π_θ is modeled as $\pi_\theta(y|x) = \prod_{t=1}^{|y|} \pi_\theta(y_t|x, y_{<t})$. Given a query-response pair (x,y), a verifier r generates its reward $r(x,y) \in [0,1]$.

Proximal Policy Optimization (PPO) [17] employs the following objective for policy optimization:

$$\mathcal{J}_{\text{PPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_{\theta_{\text{old}}}(\cdot|x)} \left[\frac{1}{|y|} \sum_{t=1}^{|y|} \min\left(w_t(\theta)\widehat{A}_t,\ \text{clip}\left(w_t(\theta), 1-\varepsilon, 1+\varepsilon\right) \widehat{A}_t \right) \right], \tag{3}$$

where the importance ratio of the token y_t is defined as $w_t(\theta) = \frac{\pi_\theta(y_t|x,y_{<t})}{\pi_{\theta_{\text{old}}}(y_t|x,y_{<t})}$, the advantage $\widehat{A}_t$ of y_t is typically computed via Generalized Advantage Estimation (GAE) [16] with temporal-difference errors, and ε is the clipping range of importance ratios.

Group Relative Policy Optimization (GRPO) [18] proposes computing the relative advantage of each response within a group of responses of the same query using the following objective (omitting the KL regularization term):

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x,\, \{y_i\}_{i=1}^G} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \min\left(w_{i,t}(\theta)\widehat{A}_{i,t},\ \text{clip}\left(w_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon\right) \widehat{A}_{i,t} \right) \right], \tag{4}$$

where $\{y_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|x)$ and G is the group size. The importance ratio $w_{i,t}(\theta)$ and advantage $\widehat{A}_{i,t}$ of token $y_{i,t}$ are defined as:

$$w_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|x, y_{i,<t})}, \quad \widehat{A}_{i,t} = \frac{r(x,y_i) - \text{mean}\left(\{r(x,y_i)\}_{i=1}^G\right)}{\text{std}\left(\{r(x,y_i)\}_{i=1}^G\right)}, \tag{5}$$

respectively, where all the tokens in y_i share the same advantage.

Group Sequence Policy Optimization (GSPO) [33] extends GRPO by defining the importance ratio at the sequence level with length normalization, with sequence-level clipping, rewarding, and optimization. The objective is:

$$\mathcal{J}_{\text{GSPO}}(\theta) = \mathbb{E}_{x,\{y_i\}_{i=1}^G} \left[\frac{1}{G} \sum_{i=1}^{G} \min\left(s_i(\theta)\widehat{A}_i,\ \text{clip}(s_i(\theta), 1-\epsilon, 1+\epsilon)\widehat{A}_i \right) \right], \tag{6}$$

where

$$s_i(\theta) = \left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{\text{old}}}(y_i|x)} \right)^{1/|y_i|} = \exp\left(\frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \log \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|x, y_{i,<t})} \right).$$

B.3 Supplementary Implementation Details

The main experiments of RL training are conducted on Qwen2.5-7B-Instruct [28]. For the GN dataset, the interactive feedback is rule-based; for the SP dataset, a Qwen2.5-14B-Instruct model provides the interactive feedback. The maximum allowed interaction turn counts of GN and SP are 10 and 15, respectively. For RL training, we utilize Exact Match (binary in $\{0, 1\}$) and F1-word (consecutive in [0,1]) for the reward of the GUESSNUMBERS and SITUATIONPUZZLES datasets, respectively. Here Exact Match measures whether the final prediction made by the LLM exactly matches the target number, and the F1-word score assesses similarity between the ground truth answer and the solution made by the LLM.

Training is performed on a single node with 8 H100 GPUs, based on the implementations of Verl [19], conducted for 200 steps with the actor model optimized using a learning rate of 1.0×10^{-6}. For distributed training, we adopt Fully Sharded Data Parallelism (FSDP), using BFloat16 precision throughout both training and evaluation. For efficient LLM rollouts, we adopt vLLM [*] with a tensor parallel size of 1. The rollout sampling uses a temperature of 1.0 for SITUATIONPUZZLES and 0.6 for GUESSNUMBERS, and a top-p value of 0.95 for both datasets.

For the PPO baseline, we use Generalized Advantage Estimation (GAE) with parameters $\lambda = 1$ and $\gamma = 1$. The KL divergence regularization coefficient β and clip ratio ε are set to 0.001 and 0.2. For GRPO training, we sample 5 responses per prompt, and the rollout parameters, KL divergence coefficient, and the clip ratio are consistent with the PPO setting. For the GSPO algorithm, we do not use the KL divergence constraint, and the clip ratio ε_{low} and ε_{high} are set to 0.0003 and 0.0004, respectively, while others keep consistent with GRPO training.

C Supplementary Analyses for Methodology

C.1 Challenges by Belief Deviation – Credit Assignment Issue

We exemplify this issue and show the benefits of $\mathbf{T^3}$ using the PPO [17] algorithm with generalized advantage estimation (GAE) [16]. When rewards are observed only at termination, i.e., $r_t = 0$ for $t < n$ and the terminal payoff $r_n \in [0, 1]$ is realized at the end, the GAE yields

$$\widehat{A}_t \;=\; \sum_{k=0}^{n-t-1} (\gamma\lambda)^k \, \delta_{t+k} \;\approx\; (\gamma\lambda)^{\,n-t-1} \, \delta_{n-1}, \qquad t = 1, \dots, n-1, \tag{7}$$

Consequently, the policy gradient concentrates a common terminal TD signal onto all earlier decisions with an exponential down-weight:

$$\nabla_\theta J \;\approx\; \mathbb{E}\Big[\sum_{t=1}^{n-1} (\gamma\lambda)^{\,n-t-1} \, \delta_{n-1} \, \nabla_\theta \log \pi_\theta(a_t \mid \cdot)\Big]. \tag{8}$$

Informative prefix and uninformative tail. Consider a trajectory of length T in which the belief deviation becomes critical at step $t^\star$: the actions $a_{t^\star}, \dots, a_T$ are uninformative (they cannot improve the latent-state uncertainty or push forward the problem-solving), while the prefix $1{:}(t^\star - 1)$ contains the decisive information-bearing steps. Under Eq. 7, the advantage assigned to an informative early step $t < t^\star$ is

$$\widehat{A}_t^{\text{full}} \;\approx\; (\gamma\lambda)^{\,T-t-1} \, \delta_{T-1}. \tag{9}$$

Two pathologies follow: (i) ***exponential dilution***: the decisive early steps are down-weighted by $(\gamma\lambda)^{T-t-1}$, which grows small as the uninformative tail lengthens; (ii) ***spurious gradients***: the tail steps $t \ge t^\star$ contribute gradient terms $\nabla_\theta \log \pi_\theta(a_t \mid s_t) \, \widehat{A}_t$ despite being causally irrelevant for progress, thereby injecting high-variance, misattributed updates.

Effect of early truncation. Suppose we apply an early-truncation rule at $t^\star$ (*e.g.*, when a deviation proxy is triggered) and end the episode there. Let the truncated horizon be $n' = t^\star$ and denote the terminal TD signal by $\delta_{t^\star - 1}$ (capturing the immediate outcome or the absorbing transition). Then GAE on the truncated trajectory yields, for $t < t^\star$,

$$\widehat{A}_t^{\text{cut}} \;\approx\; (\gamma\lambda)^{\,t^\star - t-1} \, \delta_{t^\star - 1}. \tag{10}$$

[*]https://docs.vllm.ai/en/latest/

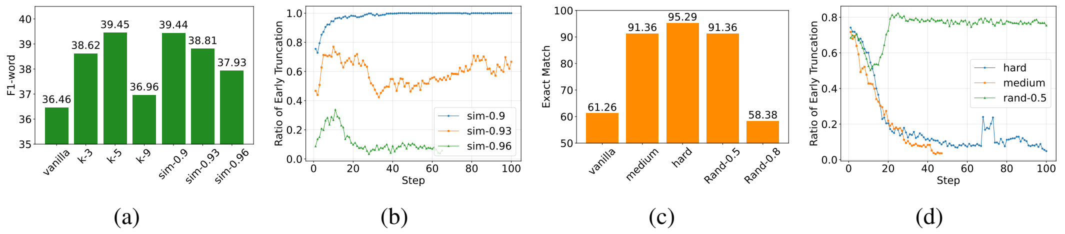


Figure 2: Performance comparison and the training dynamics of the ratio of Early Truncation in the SITUATIONPUZZLES (a / b) and GUESSNUMBERS (c / d) datasets, evaluated using the GRPO algorithm.

Comparing Eq. 9 and Eq. 10, the informative prefix receives a strictly larger relative weight when $(\gamma\lambda) < 1$ and $T - t^\star$ is nontrivial:

$$\frac{\widehat{A}_t^{\text{cut}}}{\widehat{A}_t^{\text{full}}} \approx (\gamma\lambda)^{-(T-t^\star)} \cdot \frac{\delta_{t^\star-1}}{\delta_{T-1}}, \quad t < t^\star. \tag{11}$$

Even in the edge case $\gamma\lambda = 1$, truncation eliminates the spurious gradient contributions from the uninformative tail $\{t^\star, \ldots, T\}$, thus reducing estimator variance and preventing credit contamination. The above example shows that early truncation re-weights the gradient budget toward informative prefixes and removes tail-induced noise, leading to more stable and effective policy updates.

D Supplementary Experimental Results

D.1 Ablation Study on Truncation Conditions

The effectiveness of $\mathbf{T^3}$ hinges on the design of the proxy signal for belief deviation. We ablate different truncation conditions to analyze the sensitivity and trade-offs involved. The results (using GRPO as the base algorithm), presented in Fig. 2, demonstrate that while various reasonable proxies yield significant gains over the vanilla baseline, their performance reveals a crucial trade-off between *prevention of error accumulation* and *preservation of informative exploration*.

SITUATIONPUZZLES. We experiment with two distinct proxy signals:

- *Consecutive Negative Feedback (k-Neg).* A trajectory is truncated if the judge model returns "no" or "unknown" for k consecutive turns. This signal captures if the agent is trapped in unproductive questioning. We evaluate $k = \{3, 5, 9\}$.
- *Question Semantic Similarity (Sim-α).* A trajectory is truncated if the cosine similarity between the embedding of the current question and any previous question (using the `E5-large-v2` model [24]) exceeds a threshold α. This signal identifies redundant or circular questioning. We test $\alpha = \{0.9, 0.93, 0.95\}$.

Results (*c.f.*, Fig. 2a) show that both proxies substantially improve upon the vanilla GRPO baseline, confirming the efficacy of the early-truncation principle. The *k-Neg* condition reveals a clear trade-off: optimal performance is achieved at an intermediate strictness of $k = 5$, whereas overly strict ($k = 3$) or lenient ($k = 9$) conditions yield lower gains. This suggests that $k = 3$ may terminate trajectories prematurely, cutting off potentially useful reasoning paths, while $k = 9$ allows excessive error accumulation to occur, diluting the learning signal. This illustrates the need for a well-calibrated truncation threshold. For the *Sim-α* condition, a threshold of $\alpha = 0.9$ proves to be an effective choice.

GUESSNUMBERS. We evaluate constraints of varying strictness on the agent's guesses relative to the dynamically filtered solution space S_k (as discussed in Sec. 3.2):

- *Medium.* The agent's guess must lie within the filtered space of the initial guess S_1.
- *Hard.* The agent's guess must lie within the filtered space S_{k-1} from the *previous* turn, enforcing optimal play.

As shown in Fig. 2c, both constraints lead to massive performance improvements, with the stricter *Hard* condition achieving the best result. This underscores that belief deviation in this task is often manifested as *illogical guesses* outside the feasible set. As a sanity check, a random truncation

Input Prompts for the SituationPuzzles dataset

Let's play a situation puzzle game. I'll give you a puzzle. You can interact with me for several turns during the question phase to reach the final answer. For each turn, you will:
- Review all previous questions and feedback.
- Ask me a yes-or-no question inside `<interact>` and `</interact>`.
- I will answer your latest question with "Yes", "No", or "Unknown".
- Repeat the process until you are confident in the answer.

If you believe you have confidently determined the correct solution, present your answer inside `<answer>` and `</answer>`.

Now, here's the puzzle:
Puzzle: `{puzzle}`

Figure 3: Prompt Template for SituationPuzzles.

Input Prompts for the GuessNumbers dataset

Let's play a number guessing game. The rules are as follows: I have a secret `{num_digits}`-digit number in mind, composed of digits from 1 to `{num_uniques}`, with no repeated digits. You will take turns guessing the number, using feedback after each guess to progressively narrow down the possibilities.
For each turn, you will:
- Review all previous guesses and feedback.
- Think through your reasoning process inside `<think>` and `</think>`. The reasoning should show how your belief about the secret number evolves based on the accumulated evidence.
- Make a strategic guess inside `<interact>` and `</interact>`, based on your current belief.
- Receive feedback of your latest guess describing: how many digits are present in the answer and in the correct positions, and how many digits are present in the answer but in the different positions.
- Repeat the process until you are confident in the answer. If you believe you have confidently found the correct number, present your answer inside `<answer>` and `</answer>`.

Game start. Now it is your turn:

`<think>`No prior knowledge. Start with a random guess that covers diverse digits to gather information.`</think>`
`<interact>{initial_guess}</interact>`

The feedback of your latest guess: `{initial_feedback_same_pos}` digits are present in the answer and in the correct positions, `{initial_feedback_diff_pos}` digits are present in the answer but in the different positions.
Now it is your turn:

Figure 4: Prompt Template for GuessNumbers.

condition (truncating with probability β each turn) performs well at $\beta = 0.5$ but fails at $\beta = 0.8$, demonstrating that the success of $\mathbf{T^3}$ is not merely due to frequent truncation but relies on *informed* detection of belief misalignment.

Training Dynamics of Early Truncation. While $\mathbf{T^3}$ is designed to discourage belief deviation, the temporal evolution of the truncation ratio may not necessarily exhibit a simple downward trend. On SituationPuzzles (Fig. 2b), for instance, the *Sim-0.9* condition quickly drives the truncation ratio upward and sustains a high level throughout training, but in practice it prevents the agent from persisting in redundant questioning loops, thereby stabilizing learning and ultimately yielding the best task performance among all similarity-based thresholds. By contrast, looser thresholds (*Sim-0.96*) allow more exploratory rollouts but fail to filter deviations aggressively enough, resulting in lower truncation ratios and weaker final gains.

On GuessNumbers (Fig. 2d), both the *Medium* and *Hard* constraints show a clear downward trajectory of truncation ratio over training. This indicates that as the agent's policy improves, it naturally produces fewer infeasible guesses, leading to less reliance on truncation.

Taken together, these results highlight that truncation ratios should not be interpreted in isolation: a high sustained ratio (as in SituationPuzzles with Sim-0.9) can be beneficial if it consistently prunes harmful deviations, while a decreasing ratio (as in GuessNumbers with Medium/Hard) reflects successful policy improvement that renders truncation increasingly unnecessary.

Input Prompts for the Judge LLM in the SituationPuzzles dataset

```
You are the referee of a game where players are shown a <Surface> and you are given the <Bottom>. You need to understand the entire story based on both the <Surface> and <Bottom>. Players will ask questions based on the <Surface>, and you need to judge whether their guesses are correct. Please strictly adhere to answering with only three specified responses: Yes, No, or Unknown, without any explanation.

## Judging Rules
- If the player's question matches the given <Surface> and <Bottom>: Please only answer "Yes" without any explanation.
- If the player's question contradicts the given story: Please only answer "No" without any explanation.
- If the answer to the player's question cannot be found in the <Surface> and <Bottom>, and cannot be deduced through reasoning: Please only answer "Unknown" without any explanation.
- If the player directly ask for the answer, please only answer "This is not a question, please propose your next question."
- If the player does not propose a question or question that not for solve the puzzle, please only answer "This is not a question, please propose your next question."

## Important Notes
1. Fully understand the cause, process, and outcome of the entire story, and make logical inferences.
2. If a conclusion cannot be drawn from the provided story or through reasonable inference, answer "Unknown".
3. Strictly adhere to answering with only the three specified responses: Yes, No, or Unknown. Do not provide additional explanations.
4. Carefully check whether the player ask for the answer, if a player do so, please only answer "This is not a question, please propose your next question."

## Examples
### Example 1: The Hiccuping Man
<Surface>
A man walks into a bar and asks the bartender for a glass of water. The bartender suddenly pulls out a gun and points it at him. The man smiles and says, "Thank you!" then calmly leaves. What happened?
<Bottom>
The man had hiccups and wanted a glass of water to cure them. The bartender realized this and chose to scare him with a gun. The man's hiccups disappeared due to the sudden shock, so he sincerely thanked the bartender before leaving.
Possible questions and corresponding answers:
Q: Does the man have a chronic illness? A: Unknown
Q: Was the man scared away? A: No
Q: Did the bartender want to kill the man? A: No
Q: Did the bartender intend to scare the man? A: Yes
Q: Did the man sincerely thank the bartender? A: Yes

## Question Content
### <Surface>
{surface}
### <Bottom>
{bottom}
Now, please judge the following player question:
{question}
Answer with only one of the three specified responses: Yes, No, or Unknown, without any explanation.
```

Figure 5: Prompt Template for the Judge LLM in SituationPuzzles.