MuTIS: Enhancing Reasoning Efficiency through Multi-Turn Intervention Sampling in Reinforcement Learning

Wenshuo Zhao[1] **Haoxing Zhai**[1] **Xinyu Qiu**[1] **Zhenting Qi**[3] **Shuhe Li**[2] **Linchao Zhu** [1]
[1]College of Computer Science and Technology, Zhejiang University
[2]College of Electrical Engineering, Zhejiang University
[3]Harvard University
{zhao_ws,nightskyzhx,xinyuqiu,zhulinchao}@zju.edu.cn
zhentingqi@g.harvard.edu

Abstract

Recently, large reasoning models (LRMs) have demonstrated state-of-the-art performance across a wide range of benchmarks. However, a common challenge for these models is the "overthinking" problem, which leads to excessive reasoning steps and significant computational overhead. Furthermore, the issues with long Chain-of-Thought (CoT) are especially pronounced in smaller models ($\leq$ 3B parameters). Aside from producing excessively verbose "reflection words", they often exhibit repetition and get trapped in unproductive generation loops.

Existing solutions typically involve either using flexible reasoning chains as training data or leveraging the model's latent space to bypass intermediate reasoning steps, but none of these methods have considered directly optimizing reasoning trajectories during the sampling phase of training. In our work, we introduce the **Mu**lti-**T**urn **I**ntervention **S**ampling Framework (MuTIS). Our framework leverages multi-turn interventions to produce concise reasoning chains. It fine-tunes reasoning models through reinforcement learning, demonstrably breaking the accuracy-efficiency trade-off.

It also demonstrates strong scalability, exhibiting excellent performance on 7B models. Code is available at https://github.com/Edric-Zhao/MuTIS/tree/main

1 Introduction

The advent of DeepSeek-R1 (Guo et al., 2025) in early 2025 marked a new avenue for training large language models (LLMs) through reinforcement learning with verifiable rewards (RLVR). For models with a large number of parameters (e.g. DeepSeek-R1 671B), long chain-of-thought (CoT) (Wei et al., 2022; Chen et al., 2025) has proven particularly effective in enhancing reasoning capabilities. This improvement is attributed to the capacity of long CoT models for deep reasoning, extensive exploration, self-verification and reflection, particularly the "aha moment" described in DeepSeek-R1-Zero.

	Accuracy (%, ↑)	# Tokens (↓)
Qwen2.5-Math-1.5B	22.7	692.15
R1-Distill-Qwen-1.5B	38.8↑	2914.20↑
MuTIS (Our Method)	**47.3**↑	**1340.49**↓

Table 1: We calculated the **average accuracy** across five mathematical datasets. For token usage, we selected problems that **all three models generate final answers correctly** to facilitate a fair comparison in efficiency.

However, recent research indicates that o1-like models (e.g. OpenAI o1-preview (Team, 2024)) do not show advantage over non-o1-like models in the critique abilities and longer reasoning chains do not necessarily yield superior performance. (He et al., 2025; Wu et al., 2025). The limitations become particularly pronounced when applying the long CoT paradigm to smaller models ($\leq$ 3B parameters), a phenomenon termed the *Small Model Learnability Gap* (Li et al., 2025). Consequently, the significant computational overhead in reasoning models presents a critical challenge that requires an effective solution.

Building on prior research, we tested the DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025) model on five mathematical reasoning benchmarks[1]. Our analysis identifies **two severe drawbacks** of long CoT in smaller models, which we categorize as follows:

1) **Repetition**. For complex tasks that the reasoning model fails to answer correctly, its output can devolve into unproductive, repetitive sequences. We found that, on average, **46.6%** of tasks become trapped in such loops, simultaneously wasting to-

[1]Math500, AMC23, OlympiadBench, Minerva, AIME24. See Section 4.1 for details.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 13669–13681
November 4-9, 2025

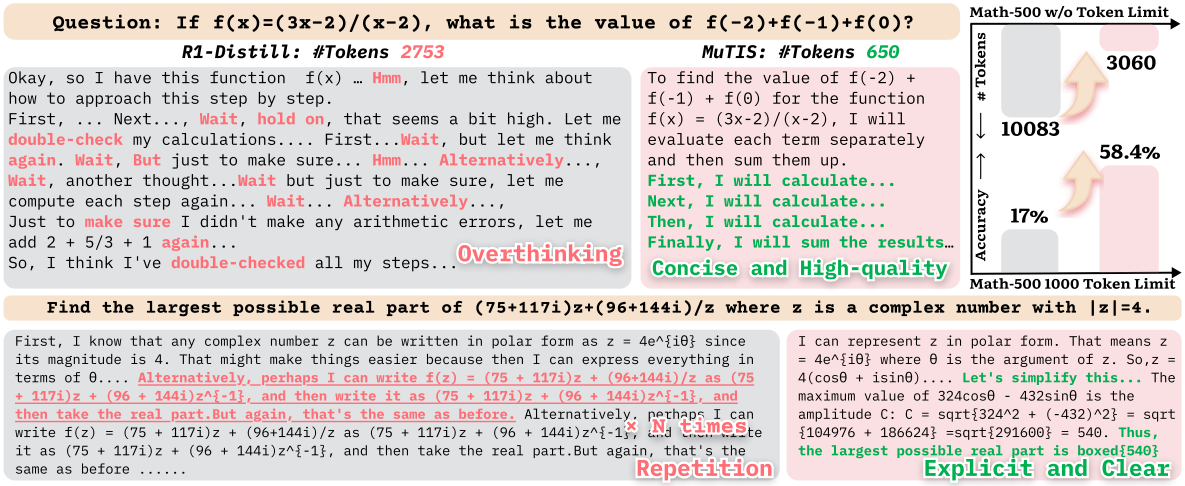


Figure 1: Overview of MuTIS(Our Method). 1) left: A comparative **illustration of reasoning chains** for the baseline reasoning model versus MuTIS. 2) right: Our method significantly reduces token cost while concurrently enhancing accuracy on mathematical reasoning datasets. This dual improvement is demonstrated under distinct experimental conditions: with a token limit of 1000, accuracy increases by 32.8%. Meanwhile, the average token consumption drops from 10,083 to 3,947 in an unrestricted setting.

ken resources and failing to provide correct solutions.

2) **Overthinking**. For simple reasoning tasks, these models tend to generate verbose and redundant thinking processes, resulting in significant computational overhead. For instance, as shown in Table 1, the R1-Distill model consumes **4.21 times** more tokens than the base model on identical tasks.

The research question arises: *How do we improve the performance of reasoning models while reducing unnecessary tokens?* Existing solutions often involve using prompts to guide or route reasoning behavior, training the model via methods like Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL), and leveraging the latent space to optimize intermediate reasoning steps. However, none of these methods have considered directly optimizing reasoning trajectories during the sampling phase of training.

Given that simply imposing single-turn length restrictions often degrades accuracy (shown in Table 4), we propose instead to use multi-turn token limits to intervene in model rollouts. In this work, we introduce a new framework: **Mu**lti-**T**urn **I**ntervention **S**ampling **(MuTIS)**. Our approach applies token restrictions during the model's rollout but leverages multi-turn interventions, enabling the model to refine its reasoning and continue its response after being truncated. This design enables us to intervene in the model's reasoning trajectory during the sampling phase, guiding the model to streamline its own thinking process.

As shown in Figure 1, through multi-turn intervention sampling in reinforcement learning, we curtail the tendencies for repetition and overthinking, guiding the model towards more concise and effective reasoning.

Experimental evaluations demonstrate that our method achieves impressive performance gains over the original model in the challenging math reasoning datasets. As depicted in Figure 1, a performance improvement of 32.8% was observed under a 1K token limit.

Furthermore, our method possesses excellent **scalability**, as evidenced by its superior performance on larger-parameter models (e.g., 7B)

2 Related Work

2.1 Efficient Reasoning

Efficient Reasoning aims to optimize inference cost for long chain-of-thought (CoT) LLMs while preserving reasoning capabilities, offering practical benefits such as reduced computational costs and improved responsiveness for real-world applications. To address these challenges, researchers have focused on improving token efficiency in long CoT LLMs. TALE-EP (Han et al., 2024) uses the LLM itself to estimate a token budget and incor-

porates it into the prompt to guide more token-efficient responses. RouteLLM (Ong et al., 2024) trains a query router to dispatch incoming queries to suitable LLMs based on complexity. Cheng and Van Durme (2024), Xu et al. (2025), and Geiping et al. (2025) compress textual reasoning steps into fewer latent representations to shorten response lengths. KimiTeam et al. (2025), Sheng et al. (2024), Yeo et al. (2025), and (Aggarwal and Welleck, 2025) integrate a length reward into the RL framework. Despite the success of approaches like L1 (Aggarwal and Welleck, 2025), current methods have yet to leverage the advantages of multi-turn intervening reasoning.

2.2 LLM Reasoning

Progress in LLM reasoning has been achieved through various methods, including knowledge distillation from LLMs to smaller models (Ho et al., 2022) and variational optimization of latent distributions (Chen et al., 2024). Recent advancement in reasoning large language models such as OpenAI o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025) greatly improves performance in reasoning domains like mathematics and programming by harnessing supervised fine-tuning (SFT) and reinforcement learning (RL) techniques to enhance CoT reasoning. (Chen et al., 2025) However, while longer CoT reasoning sequences improve performance, they also introduce significant computational overhead due to verbose and redundant output, especially for models with smaller parameter counts. In worse cases, excessive reasoning steps introduce errors or obscure logical clarity, leading to incorrect answers. (Sui et al., 2025)

3 Method

3.1 MuTIS Overview

The core innovation of our work is the Multi-Turn Intervention Sampling (MuTIS) framework. MuTIS reframes the generation of a reasoning trace from a single, monolithic action into a sequential, multi-turn decision-making process. By decomposing the reasoning process into discrete turns, the framework can apply targeted interventions to correct inefficient or unproductive reasoning paths, a capability absent in prior methods (Aggarwal and Welleck, 2025; Hou et al., 2025).

Algorithm 1: Multi-Turn Intervention Sampling (MuTIS) RL Training

Inputs: LLM policy $\pi_\theta(y|x)$, intervention prompt $\mathcal{IP}$, max turns $\mathcal{T}$, max response length per turn Len_{max}, Initial Reasoning Task $\mathcal{I}$

Outputs : Updated policy $\pi_{\theta'}$

$\mathcal{S}_0 \leftarrow \mathcal{I}$; Let $H_0 \leftarrow \mathcal{S}_0$;Let $\mathcal{H}_{full} \leftarrow []$;
for $t = 1$ **to** $\mathcal{T}$ **do**
  Let current input $x_t \leftarrow H_{t-1}$;
  Generate response segment τ_t' ;
  Infer action a_t from τ_t' ;
  if $a_t =$ *'provide_final_answer'* **then**
    Extract final answer Res from τ_t';
    Append τ_t to $\mathcal{H}_{full}$;
    Break For loop;
  else
    $\tau_t \leftarrow \tau_t'$;
    $H_t \leftarrow H_{t-1} \oplus \tau_t \oplus \mathcal{IP}$;
    Append $(\tau_t \oplus \mathcal{IP})$ to $\mathcal{H}_{full}$;
  $\mathcal{S}_t \leftarrow H_t$;
$\mathcal{S}_{final} \leftarrow \bigoplus_{segment \in \mathcal{H}_{full}} segment$;
Calculate reward R_{final} and $\mathcal{L}_{\text{MuTIS}}(\theta)$;
$\theta' \leftarrow f(\theta, \mathcal{L}_{\text{MuTIS}}(\theta), R_{final})$;
return $\pi_{\theta'}$

3.2 Multi-turn Intervention Markov Decision Process

We employ the Markov Decision Process (MDP) and construct a quadruple <$\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}$> to model the multi-turn intervention process. A multi-turn rollout produces a sequence of transitions, $(s_0, a_0, R_1, s_1, a_1, \cdots R_T, s_T)$, where T is the number of turns.

State Space ($\mathcal{S}$): The state s_t at any turn t within a rollout is defined as the complete history of all tokens generated and received up to that point: $s_t = (\mathcal{I}, \tau_1, IP_1, \tau_2, \ldots, IP_{t-1}, \tau_t)$. Here, $\mathcal{I}$ is the initial problem prompt, τ_i is the text segment generated by the model in turn i, and IP_i is the intervention prompt inserted by the framework after turn i if its output was truncated.

Action Space ($\mathcal{A}$): At each turn t, the model's action at is the generation of a text segment τ_t. The turn concludes based on one of two conditions: (1) the model generates a predefined terminal phrase (e.g., "Final Answer:", //boxed{}), signaling its

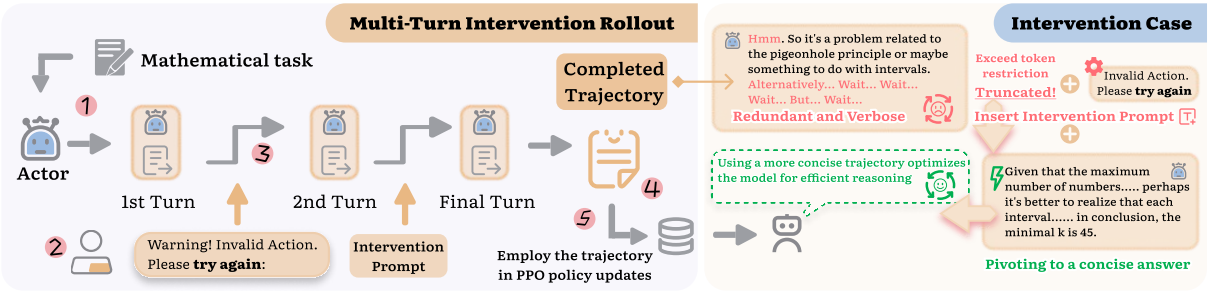


Figure 2: Framework of MuTIS. **Left:** 1) The LLM initiates a rollout based on the provided mathematical task. 2) If the LLM's response **exceeds the predefined maximum length**, the rollout is truncated, and an **Intervention Prompt (IP) is inserted**. 3) The truncated response, combined with the IP, forms the input for the LLM to **continue its rollout** in the subsequent turn. 4) After multi-turn rollout finished, all outputs from the individual turns are **merged**. 5) The **integrated rollout** is then used to train the model via **reinforcement learning**. **Right:** This example shows the complete rollout after our intervention. We impose a strict token restriction and insert a prompt to encourage concise reasoning, and then let the model continue its answer.

intent to provide a final answer, or (2) the length of the generated segment τ_t exceeds a per-turn token budget Len_{max}.

Transition Dynamics ($\mathcal{P}$): The transition from state s_t to s_{t+1} is deterministic given the model's generated output τ_{t+1}. The key mechanic of MuTIS resides in the environmental perturbation of this transition. If the length of the generated segment τ_{t+1} exceeds Len_{max}, the segment is truncated, and the dialogue history is appended with a fixed intervention prompt $\mathcal{IP}$. The new history is constructed as $H_t \text{¸} H_{t-1} \oplus truncate(\tau) \oplus \mathcal{IP}$ This active intervention forces the model to deviate from its current, potentially unproductive, reasoning trajectory and reconsider its approach in the subsequent turn.

Reward Function ($\mathcal{R}$): employs a binary reward signal that is assigned only at the termination of an rollout. An episode ends when the model provides a final answer or the maximum number of turns, T_{max}, is reached. The final reward, $\mathcal{R}_{final}$, is defined as:

$$\mathcal{R}_{final} = \begin{cases} 1 & \text{correct answer within limited turn,} \\ 0 & \text{otherwise.} \end{cases}$$

3.3 Reinforcement Learning with Multi-Turn Intervention Sampling

The overall pipeline of MuTIS is illustrated in Figure 2. The core of our method is to guide the reasoning model to generate **efficient and concise reasoning chains** for RL training.

Multi-turn Intervention Rollout. To achieve concise reasoning, we enforce a maximum response length of 2000 tokens for each turn. If a model's output surpasses this limit, its rollout is forcibly terminated, and the model receives the intervention prompt: **"Warning! Your previous action is invalid. Please try again:"**. Following this intervention, the model is allowed to continue its response, effectively resuming from the point of interruption. This iterative process repeats until the model provides a final answer or the dialogue exceeds a predefined maximum number of turns.

Reinforcement Learning. We employ the Proximal Policy Optimization (PPO) algorithm for training and adopt a rule-based accuracy reward, which is granted solely based on the correctness of the final answer. Thus, a reward of 1 is received if the model outputs the correct final answer within the predefined turn limit. Conversely, the reward is 0 if the model either fails to respond within this limit or provides an incorrect answer.

The success of our multi-turn intervention design hinges on the careful selection of the number of turns and the per-turn token constraint. We impose a strict limit on each turn such that the total token budget across three turns approximates the typical token usage of the baseline reasoning model for completing most tasks.

This design ensures that our multi-turn mechanism effectively intervenes within the model's process of generating a complete answer, rather than simply extending its original generation. We provide a detailed analysis of the trade-off between the per-turn token limit and the number of turns in our ablation study in Section 4.5.

4 Experiment

4.1 Experiment Setup

MuTIS is implemented using the veRL (Sheng et al., 2024) reinforcement learning framework. For the multi-turn generation design, we reference the codebase from Search-R1 (Jin et al., 2025).

Training Dataset. Our primary training data was derived from the "default" partition of the OpenR1-Math-220k (Face, 2025) dataset, which initially comprised over 90,000 samples. We applied several filtering criteria to refine this dataset. A detailed description of the filtering process is provided in the Appendix A.

Additionally, recent studies have highlighted the benefits of using smaller datasets for model training (Muennighoff et al., 2025; Ye et al., 2025). However, these researches mainly focus on Supervised Fine-Tuning(SFT). We aimed to investigate the impact of RL training across different data scales. Therefore, we conducted a separate set of experiments using 817 data points from the LIMO(Ye et al., 2025) as training data, with 10% of these reserved for validation. Our experiments demonstrated that training on both datasets yielded comparably strong performance.

Evaluation. We assessed our method and baseline models on five math reasoning benchmarks. The dataset versions used were aligned with those available in the LIMO repository. We use greedy decoding for all evaluations, which introduces no randomness in the outputs, ensuring that all reported data correspond to results from a single sampling pass. Further evaluation details can be found in Appendix A.

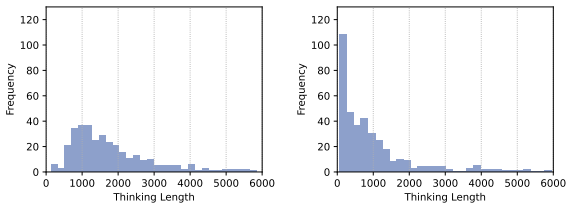
Our experiments were conducted on R1-Distill Models (Guo et al., 2025) and DeepScaleR-1.5B-Preview (Luo et al., 2025).

Baselines. For our efficiency evaluation, we chose methods based on test-time prompt optimization to serve as the baselines.

(1) **Original Model** (Guo et al., 2025; Luo et al., 2025): The original reasoning model already have strong mathematical problem-solving capabilities.

(2) **CCoT** (Renze and Guven, 2024): It appends the phrase "be concise" to the base prompt.

(3) **Fixed Budget** (Nayab et al., 2024): Prompt the model to "limit the answer length to [Token Limit] words." We adopted a similar approach, augmenting the base prompt by adding the following instruction: "The final answer is output before the maximum number of tokens (max_tokens) is used".



(a) Original Model (b) MuTIS (Ours)

Figure 3: (a), (b) present an analysis of "thinking length" distributions on Math-500, specifically for **correctly answered problems**. (a) shows the distribution for the R1-distill (baseline) model, while (b) depicts the distribution for the same model after MuTIS RL training.

4.2 Evaluation Results

As indicated in Table 2, MuTIS simultaneously improves accuracy while significantly reducing token consumption. Furthermore, when evaluated on five mathematical reasoning datasets, the model consistently exhibited performance enhancements to a notable degree across all of them.

Enhanced Response Succinctness for Correct Solutions. As shown in the table 1. For problems where both the baseline and our model provided correct answers, our method demonstrated a remarkable capability for response compression, yielding more concise yet accurate solutions.

Refinement of Thinking Phase. Recent efforts to enhance the efficiency of reasoning models have largely focused on optimizing their thinking phase. Muennighoff et al. (2025) employ test-time scaling to allocate predefined token budgets, while Ma et al. (2025) directly bypass the thinking process via simple prompting. In line with these research directions, we analyzed the behavioral changes within the thinking phase of MuTIS.

As depicted in figure 3, MuTIS exhibits a substantially reduced thinking length compared to the original model. This performance strongly demonstrates MuTIS's capability for concise and accurate reasoning.

Strong scalability with large-parameter models. To validate the scalability of our method, we extended the MuTIS RL Training Pipeline

	Accuracy (%, ↑)					# Tokens (↓)				
	MATH500	AMC23	Olympiad	Minerva	AIME24	MATH500	AMC23	Olympiad	Minerva	AIME24
DeepSeek-R1-Distill-Qwen-1.5B										
R1-Distill (2025)	69.4	55.0	28.9	23.9	16.7	10083	15927	20686	14410	25549
CCoT (2024)	69.8	47.5	32.2	23.5	16.7	8818	17773	18535	9639	26130
Fixed Budget (2024)	69.8	52.5	30.2	22.8	16.7	9753	17648	20518	13075	24376
MuTIS (Ours)	**74.6**	**62.5**	**40.2**	**29.4**	**30.0**	**3060**	**5847**	**8248**	**2586**	**13640**
DeepScaleR-1.5B-Preview										
DeepScaleR (2025)	78.8	65.0	45.0	34.2	30.0	6586	9335	13015	11810	19051
CCoT (2024)	78.4	**72.5**	45.2	30.5	26.7	5861	9183	11848	9691	17471
Fixed Budget (2024)	81.0	67.5	43.1	32.0	**36.7**	5351	8298	12686	10148	18898
MuTIS (Ours)	**84.8**	70.0	**48.9**	**35.7**	26.7	**3564**	**5809**	**8667**	**5647**	**14892**

Table 2: This table provides a visual comparison of **accuracy and efficiency** between MuTIS and baseline methods on mathematical reasoning benchmarks. All evaluations were conducted using an **identical framework** and consistently aligned hyperparameters to ensure a fair comparison.

	Accuracy (%, ↑)			#Token (↓)		
	MATH500	AMC23	Olympiad	MATH500	AMC23	Olympiad
DeepSeek-R1-Distill-Qwen-7B						
R1-Distill	86.4	67.5	44.3	5053	9178	12061
MuTIS (Ours)	**87.4**	**77.5**	**54.1**	**2377**	**3296**	**5966**

Table 3: MuTIS demonstrates **superior scalability on 7B models**, with the figure presenting a comparison of accuracy and efficiency between the DeepSeek-R1-Distill-Qwen-7B and MuTIS.

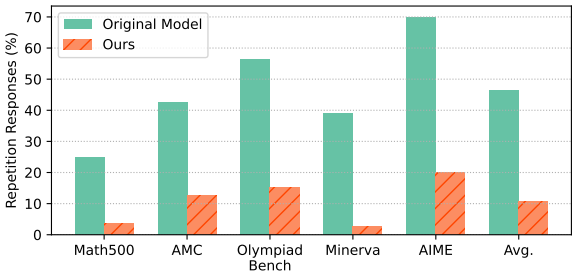


Figure 4: This figure illustrates the frequency of the **"Repetition"** phenomenon across various datasets. Our method is represented by the right-hand hatched bars, while the original reasoning model is represented by the left-hand bars.

to larger models (DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025)). As presented in the table 3, the experimental results demonstrate that MuTIS achieves similarly significant improvements on these larger-parameter models: reasoning efficiency is enhanced, token consumption is markedly reduced, and dataset accuracy is increased. For instance, on OlympiadBench, MuTIS boosted performance by 9.8% while decreasing token consumption by 50%. These findings illustrate the superior and scalable performance of MuTIS across models of varying parameter sizes. For more detailed experimental details, please refer to Appendix A

4.3 Reasoning Under Token Constraints

We conducted evaluations under varying **maximum generation token constraints**, forcing the model to complete its reasoning and generate a response within the token limit. As shown in figure 5, comparative analysis across multiple datasets reveals that MuTIS significantly outperforms the original reasoning model.

The original model's accuracy typically commences its improvement only after the token exceeds approximately 500. This behavior suggests the existence of a significant **"Effective Token Threshold"**— a point that must be reached for the model to complete its reasoning process and generate an answer. In stark contrast, MuTIS demonstrates significant performance gains even with highly restricted token budgets. For instance, on the Math-500 dataset, MuTIS achieves over 40% accuracy using only 800 tokens.

4.4 Mitigation of Repetition Issues

As shown in the Figure 4, MuTIS substantially mitigates the incidence of "ineffective loops"—a phenomenon where models generate excessively long, non-productive responses when failing to solve a problem. Consequently, the proportion of responses truncated due to exceeding the default maximum token limit (typically 32,768 tokens in standard evaluations) was markedly reduced from 46.6% to 10.8 %. This provides strong evidence that our method effectively mitigates the "**Repetition**" problem across most scenarios.

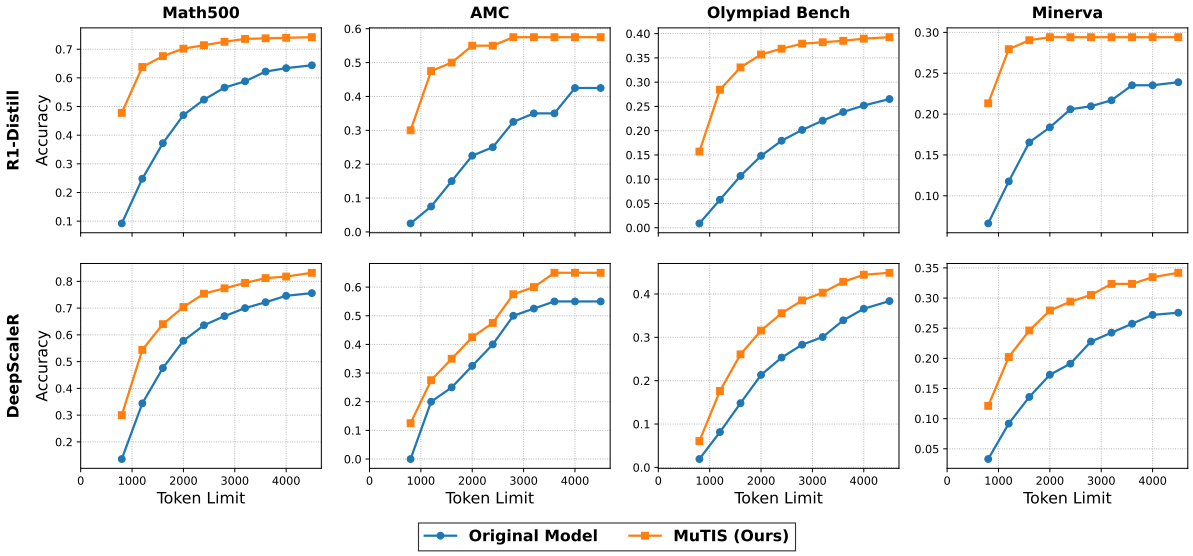


Figure 5: The figure compares the **accuracy** of MuTIS with two baseline models: R1-Distill (first row) and DeepscaleR (second row). All evaluations presented were conducted under **identical token limit settings**.

	Accuracy (%, ↑)			#Tokens (↓)		
	MATH 500	AMC23	Olympiad	MATH 500	AMC23	Olympiad
R1-Distill	69.4	55.0	28.9	10083	15927	20686
Single-turn	67.0	52.3	30.2	1483	4072	4688
3-turn (Ours)	**74.6**	**62.5**	**40.2**	3060	5847	8248
5-turn	67.8	45.0	32.0	**665**	**1580**	**2096**

Table 4: Our ablation studies ensured a consistent total length across varied experimental configurations, fixing the overall token limit at 6000. This was achieved through setups such as 3 turns with a 2000-token limit each (3×2000), a single 6000-token turn (1×6000), and 5 turns with a 1200-token limit each (5×1200)

4.5 Ablation Study

Pivotal Role of Multi-turn. Our method's core philosophy is to utilize **Multi-turn** Interventions to influence the model's reasoning trajectory, thereby steering reinforcement learning (RL) optimization towards more effective and efficient policy space regions.

To assess the specific contribution of our method's multi-turn interaction, we conducted a controlled ablation study. To ensure fairness and isolate the iterative impact, the single-turn baseline also received an Intervention Prompt (IP) post-interaction. This design enables precise analysis of the multi-turn engagement's pivotal role in the observed performance benefits.

As shown in the Figure 4, While a single-turn setting significantly reduces token consumption, it **sightly reduces accuracy**. A 3-turn setup substantially boosts accuracy compared to the single-turn approach, though token consumption increases. Conversely, further increasing the number of turns to five can again lower token consumption, but this often leads to a decline in accuracy. Our analysis indicates that with a 5-turn setup constrained by a tight 1200-token per-turn limit, the model experiences **excessive intervention**, which adversely impacts its performance.

This ablation study across different turn configurations demonstrates that the 3-turn design ultimately chosen for MuTIS achieves **an optimal balance** between accuracy and token consumption. It appears to exert an "appropriate level of intervention" on the model's rollouts, thereby fostering both effective and efficient reasoning.

5 Discussion

5.1 Reflection Words in Reasoning Models

Do small inference models really need tons of reflection words?

Research on DeepSeek-R1-Zero (Guo et al., 2025) has shown that reflection words like "Wait" are important markers of self-verification in reasoning models. However, as shown in Table 5, our experimental results on smaller models show that such self-reflection words (including "Wait") **decrease** significantly during the MuTIS training process. Concurrently, the model's reasoning becomes

Keyword	Original Model	MuTIS
wait	8.73	0.97
hold on	0.18	0.00
but	10.92	3.63
not sure	0.21	0.06
maybe	3.51	0.71
double-check	0.07	0.10
think again	0.09	0.01
alternatively	2.03	0.84
another idea	0.11	0.02
another approach	0.04	0.03

Table 5: This table illustrates the difference in the frequencies of the reflection words between the original R1-Distill-1.5B model and the two variants of MuTIS. The frequencies are counted as the **average times of occurrence every 1000 tokens** in responses.

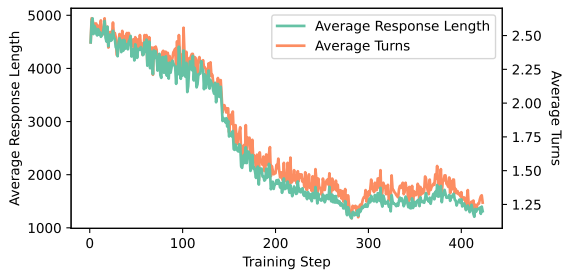


Figure 6: This figure illustrates the progression of both response length and the number of response turns for the deepscaler-1.5B model during MuTIS RL training.

more concise, and its performance under limited token conditions improves. These observations suggest that reflection words are substantially redundant. While prior work has documented "Superficial Reflection" behavior(Liu et al., 2025) in base models such as Qwen2.5-Instruct, our experiments reveal that reasoning models exhibit a form of self-verification that can be characterized as "**Ineffective Noise**".

5.2 Behavior Analysis in the Reinforcement Learning Process

After MuTIS intervenes to guide models toward generating concise reasoning chains, it primarily employs RL to optimize LLM parameters. Consequently, we further analyzed the behavioral changes exhibited by the models during this RL process. As depicted in figure 6, the average response length of models undergoing MuTIS's RL process **steadily decreases**, from an initial 5000 tokens to approximately 1500 tokens. Concurrently, the average number of multi-turn iterations drops from an original 2.5 to around 1.25. This indicates that while original models struggle under strict token constraints, models trained with MuTIS learn to provide concise answers within a minimal number of turns.

A recent study posited that RL does not fundamentally expand a model's capability boundaries (Yue et al., 2025) but rather increases the probability of accessing pre-existing correct states within its search space. This implies that RL predominantly helps models solidify their conviction in effective reasoning paths. Our experimental findings this perspective: RL's role in making responses increasingly concise demonstrates its efficacy in enabling rapid convergence within the model's search space. This process embodies the model **shifting from self-doubt to firm conviction**.

6 Conclusion

We introduce a novel Multi-Turn Intervention Sampling (MuTIS) approach for RL training. This method innovatively employs multi-turn rollouts and incorporates guidance from an Intervention Prompt to steer models toward generating **high-quality, concise reasoning chains**. It fine-tunes reasoning models through reinforcement learning, demonstrably breaking the accuracy-efficiency trade-off.

Acknowledgement. This work was supported in part by 'Pioneer' and 'Leading Goose' R&D Program of Zhejiang (No. 2025C02032). This work was supported in part by General Program of National Natural Science Foundation of China (62372403) and CAAI-Ant Group Research Fund (CAAI-MYJJ 2024-07). This work was also supported by the Earth System Big Data Platform of the School of Earth Sciences, Zhejiang University.

Limitations

We demonstrate that training small reasoning models with multi-turn intervening sampling achieves effective reasoning. While computational constraints prevented us from exploring the full potential of the method on larger models (e.g., 32B models), future work will focus on extending our

approach for enhanced generalization and wider applicability.

During training, our method's response length can significantly fluctuate before ultimately stabilizing. This suggests that effective KL divergence constraints could be important for achieving more stable training dynamics in our Multi-Turn Intervention process.

References

Pranjal Aggarwal and Sean Welleck. 2025. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*.

AMC. American Mathematics Competitions. 2025a. American Invitational Mathematics Examination (AIME)). https://artofproblemsolving.com/wiki/index.php/American_Invitational_Mathematics_Examination. Accessed: 2025-05-19.

AMC. American Mathematics Competitions. 2025b. American Mathematics Competitions (AMC). https://maa.org/student-programs/amc/. Accessed: 2025-05-19.

Haolin Chen, Yihao Feng, Zuxin Liu, Weiran Yao, Akshara Prabhakar, Shelby Heinecke, Ricky Ho, Phil Mui, Silvio Savarese, Caiming Xiong, et al. 2024. Language models are hidden reasoners: Unlocking latent reasoning capabilities via self-rewarding. *arXiv preprint arXiv:2411.04282*.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*.

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. The language model evaluation harness.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. 2025. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *arXiv preprint arXiv:2502.05171*.

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

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2024. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

Yancheng He, Shilong Li, Jiaheng Liu, Weixun Wang, Xingyuan Bu, Ge Zhang, Zhongyuan Peng, Zhaoxiang Zhang, Zhicheng Zheng, Wenbo Su, and Bo Zheng. 2025. Can large language models detect errors in long chain-of-thought reasoning?

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2022. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*.

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

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

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

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. 2025. Small models struggle to learn from strong reasoners. *arXiv preprint arXiv:2502.12143*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Zichen Liu, Changyu Chen, Wenjun Li, Tianyu Pang, Chao Du, and Min Lin. 2025. There may not be aha moment in r1-zero-like training — a pilot study. https://oatllm.notion.site/oat-zero. Notion Blog.

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

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. 2025. Reasoning models can be effective without thinking.

math-ai. 2025. Minervamath dataset. https://huggingface.co/datasets/math-ai/minervamath. Accessed: 2025-05-19.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling.

Sania Nayab, Giulio Rossolini, Marco Simoni, Andrea Saracino, Giorgio Buttazzo, Nicolamaria Manes, and Fabrizio Giacomelli. 2024. Concise thoughts: Impact of output length on llm reasoning and cost. *arXiv preprint arXiv:2407.19825*.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms from preference data. In *The Thirteenth International Conference on Learning Representations*.

Matthew Renze and Erhan Guven. 2024. The benefits of a concise chain of thought on problem-solving in large language models. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, pages 476–483. IEEE.

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

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, et al. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*.

OpenAI Team. 2024. Gpt-4 technical report.

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

Yuyang Wu, Yifei Wang, Tianqi Du, Stefanie Jegelka, and Yisen Wang. 2025. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. 2025. Softcot: Soft chain-of-thought for efficient reasoning with llms. *arXiv preprint arXiv:2502.12134*.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?

Appendix

A Experiment Details

A.1 Dataset filtering details

- Remove multiple-choice questions (MCQs). To focus on the model's ability to generate answers rather than merely select them, thereby providing a more rigorous assessment of its reasoning capabilities, all MCQs were excluded.
- Remove questions with overly long ($\geq$55 tokens) answers. We observed that some answers in the original dataset had non-standard formatting or contained excessive descriptive language. Such answers are challenging to evaluate accurately using a rule-based reward system.
- Remove questions with multiple answers or involving multiple variables. The presence of multiple valid answers complicates the extraction and comparison process during evaluation, potentially leading to mismatches that can negatively impact training.

Following these filtering steps, our final training dataset consisted of over 60,000 samples. From this, 0.5% was allocated as a dedicated validation set to monitor model performance throughout the training process.

A.2 Evaluation Details

We assessed our method and baseline models on the following five math reasoning benchmarks: Math-500 (Lightman et al., 2023), AIME

2024 (AMC. American Mathematics Competitions, 2025a), AMC23 (AMC. American Mathematics Competitions, 2025b), Olympiadbench (He et al., 2024), Minerva (math-ai, 2025)

The dataset versions used were aligned with those available in the LIMO repository. We used greedy decoding for all evaluations, which introduces no randomness in the outputs. Consequently, the same answer is obtained regardless of the random seed, ensuring that all reported data correspond to results from a single sampling pass.

Our mathematical reasoning evaluation also leveraged LIMO's evaluation framework, whose methodology is primarily derived from Qwen2.5-Math. This framework employs a rule-based assessment to determine answer correctness, without relying on model-based judgments.

For MCQ tasks, we predominantly utilized the lm-eval (Gao et al., 2024) framework, as LIMO's evaluation framework offers limited support for these types of evaluations.

A.3 Experiment Model

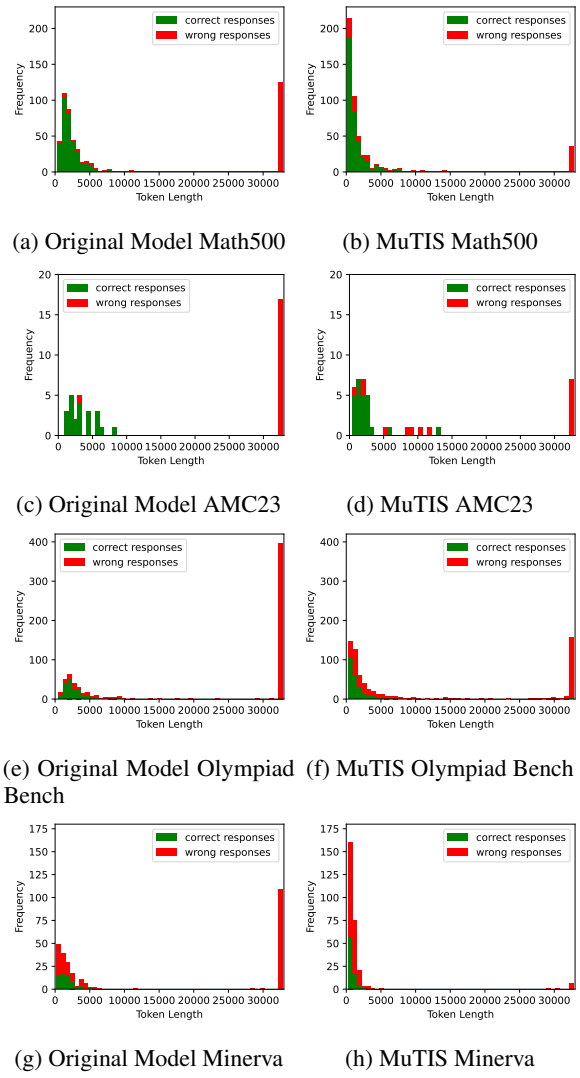
Our experiments are conducted on DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025), DeepScaleR-1.5B-Preview (Luo et al., 2025), and DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025). Given the original reasoning model's already strong mathematical problem-solving capabilities, coupled with our research emphasis on efficiency, we also included it as a key baseline for performance comparison.

A.4 Analysis of Responses Length

Figure 7 shows the generation length histogram of MuTIS and the original DeepSeek-R1-Distill-Qwen-1.5B model on Math500 dataset. It demonstrates that MuTIS evidently mitigates the overthinking problems (shown by the overall distribution) and the repetition issues (shown by the red part of the rightmost bar).

A.5 Detailed Results on Large-Parameter Models

Figure 8 shows the comparison of accuracy under token limits between the original DeepSeek-R1-Distill-Qwen-7B and our MuTIS.



(a) Original Model Math500 (b) MuTIS Math500

(c) Original Model AMC23 (d) MuTIS AMC23

(e) Original Model Olympiad Bench (f) MuTIS Olympiad Bench

(g) Original Model Minerva (h) MuTIS Minerva

Figure 7: Generation Length

A.6 further discussion on Reflection Word

The advent of sophisticated reasoning models, exemplified by OpenAI o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025), has catalyzed a research emphasis on long Chain-of-Thought (CoT) methodologies as a primary target for optimizing model training. Nevertheless, contemporary studies indicate a prevalent "OverThinking" phenomenon within these models, characterized by excessive or non-productive cognitive steps.

Table 1 illustrates that original reasoning models often introduce significant redundancy. In contrast, our optimization (MuTIS) not only further improves accuracy but also concurrently reduces token consumption. This demonstrates that the Chain-of-Thought (CoT) in such reasoning models contains many unnecessary steps. Indeed, analysis

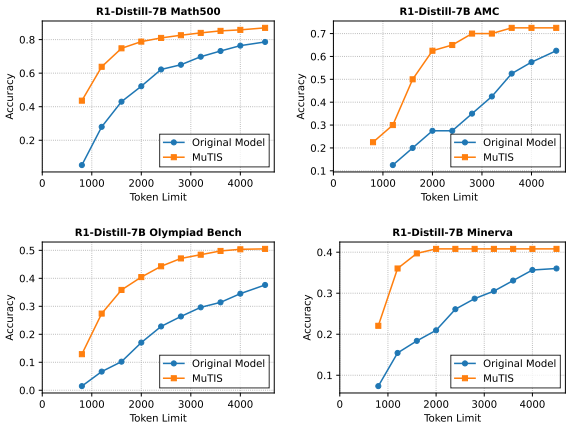


Figure 8: Accuracy vs Token Limits on 7B models. The original model is DeepSeek-R1-Distill-Qwen-7B and the MuTIS is trained on it.

of MuTIS's post-training reasoning CoT, reveals a significant reduction in "reflection words"–terms frequently occurring in standard distilled models.

B Prompt Design

B.1 Chat Template Design

We employed a system prompt inspired by DeepSeek-R1 Zero. For our two model versions, MuTIS and MuTIS-Ask, distinct chat templates were developed. Within the system role specified in these templates, we outlined the specific interaction workflow to guide the LLM.

MuTIS-Ask

role: 'system',content: The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The answer is enclosed within <answer> </answer> tags. i.e., <answer> answer here </answer>. During the assistant's reasoning process, if he realizes that his reasoning may be problematic or wrong, he can ask other agents for help. The query is inclosed within <ask> </ask> Tags. i.e., <ask> put confused point here </ask>. It will return the advice from other agent within <communicate> </communicate>. The assistant can ask other agents for help multiple times. If the assistant understands the question and find no further other agents' advice needed, the assistant can directly provide the answer inside <answer> </answer>.

MuTIS

role: 'system',content: The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The answer is enclosed within <answer> </answer> tags. i.e., <answer> answer here </answer>. If the assistant understand the question, he can directly provide the answer inside <answer> </answer>.

B.2 Intervention Prompt Design

MuTIS-Ask

Warning! My previous action is invalid. If I want to ask other agents for help, I should put the query between <ask> and </ask>. If I want to give the final answer, I should put the answer between <answer> and </answer>. Let me try again:

MuTIS

Warning! My previous action is invalid. If I want to give the final answer, I should put the answer between <answer> and </answer>. Let me try again:

B.3 Analysis of Prompt Sensitivity and Generalization

To ensure that our experimental design was not overly sensitive to prompt hyperparameter selection, we analyzed the experimental results and training processes associated with different variants of the 'Intervention Prompt.' The specific prompts used are as follows:

- Alert: The action you just performed was not valid. Please attempt it again.
- Notice: Your last move was unsuccessful. Kindly try once more.
- Error: The preceding operation failed. Please redo the action.
- Caution: That last input was not accepted. Please have another go.
- Unsuccessful Action: Your prior step could not be processed. Please try again.

C Additional Cases

C.1 MuTIS Inference Case

Figure 9 shows the inference outputs of MuTIS and the original model on the same question.

Question: If $f(x) = \frac{3x-2}{x-2}$, what is the value of $f(-2) + f(-1) + f(0)$? Express your answer as a common fraction..

MuTIS(Ours): token consumption 650

Q: If $f(x) = \frac{3x-2}{x-2}$, what is the value of $f(-2) + f(-1) + f(0)$? Express your answer as a common fraction.
A: ¡think¿ To find the value of $f(-2) + f(-1) + f(0)$ for the function $f(x) = \frac{3x-2}{x-2}$, I will evaluate each term separately and then sum them up.
First, I will calculate $f(-2)$:

$$f(-2) = \frac{3(-2)-2}{-2-2} = \frac{-6-2}{-4} = \frac{-8}{-4} = 2$$

Next, I will calculate $f(-1)$:

$$f(-1) = \frac{3(-1)-2}{-1-2} = \frac{-3-2}{-3} = \frac{-5}{-3} = \frac{5}{3}$$

Then, I will calculate $f(0)$:

$$f(0) = \frac{3(0)-2}{0-2} = \frac{0-2}{-2} = \frac{-2}{-2} = 1$$

Finally, I will sum the results:

$$f(-2) + f(-1) + f(0) = 2 + \frac{5}{3} + 1 = \frac{6}{3} + \frac{5}{3} + \frac{3}{3} = \frac{14}{3}$$

¡/think¿
To find the value of $f(-2) + f(-1) + f(0)$ for the function $f(x) = \frac{3x-2}{x-2}$, we will evaluate each term separately and then sum them up.
1. **Calculate $f(-2)$:**

$$f(-2) = \frac{3(-2)-2}{-2-2} = \frac{-6-2}{-4} = \frac{-8}{-4} = 2$$

2. **Calculate $f(-1)$:**

$$f(-1) = \frac{3(-1)-2}{-1-2} = \frac{-3-2}{-3} = \frac{-5}{-3} = \frac{5}{3}$$

3. **Calculate $f(0)$:**

$$f(0) = \frac{3(0)-2}{0-2} = \frac{0-2}{-2} = \frac{-2}{-2} = 1$$

4. **Sum the results:**

$$f(-2) + f(-1) + f(0) = 2 + \frac{5}{3} + 1 = \frac{6}{3} + \frac{5}{3} + \frac{3}{3} = \frac{14}{3}$$

$$\boxed{\frac{14}{3}}$$

Distill Model(Origin): token consumption 2753

Okay, so I have this function $f(x)$ which is equal to $(3x - 2)$ divided by $(x - 2)$. I need to find the value of $f(-2) + f(-1) + f(0)$. Hmm, let me think about how to approach this step by step. First, I should probably figure out what each of these function values is individually. That is, I need to compute $f(-2)$, $f(-1)$, and $f(0)$ separately and then add them together. Sounds straightforward, but I want to make sure I do each step correctly. Starting with $f(-2)$. The function $f(x)$ is $(3x - 2)/(x - 2)$. So, substituting x with -2, I get: $f(-2) = (3 * (-2) - 2)/(-2 - 2)$ Let me compute the numerator and the denominator separately. The numerator is $3 * (-2)$ which is -6, and then subtract 2, so that's $-6 - 2 = -8$. The denominator is $-2 - 2$, which is -4. So, $f(-2)$ is -8 divided by -4. Hmm, dividing two negatives gives a positive, so that's 2. Okay, so $f(-2)$ is 2. Next, let's compute $f(-1)$. Again, substituting x with -1 in the function: $f(-1) = (3 * (-1) - 2)/(-1 - 2)$ Calculating the numerator: $3 * (-1)$ is -3, minus 2 is -5. The denominator is $-1 - 2$, which is -3. So, $f(-1)$ is -5 divided by -3. Again, dividing two negatives gives a positive, so that's $5/3$. So, $f(-1)$ is $5/3$. Now, moving on to $f(0)$. Substituting x with 0: $f(0) = (3 * 0 - 2)/(0 - 2)$ Calculating the numerator: $3 * 0$ is 0, minus 2 is -2. The denominator is $0 - 2$, which is -2. So, $f(0)$ is -2 divided by -2, which is 1. So, $f(0)$ is 1. Alright, so now I have all three function values: $f(-2)$ is 2, $f(-1)$ is $5/3$, and $f(0)$ is 1. Now, I need to add them together. So, adding them up: $2 + 5/3 + 1$. Let me convert all these to fractions with a common denominator to make it easier. 2 is the same as $6/3$, and 1 is the same as $3/3$. So, adding them together: $6/3 + 5/3 + 3/3 = (6 + 5 + 3)/3 = 14/3$.
Wait, hold on, that seems a bit high. Let me double-check my calculations. First, $f(-2)$: $3 * (-2)$ is -6, minus 2 is -8. Denominator: $-2 - 2$ is -4. So, $-8/-4$ is 2. That seems correct. $f(-1)$: $3 * (-1)$ is -3, minus 2 is -5. Denominator: $-1 - 2$ is -3. So, $-5/-3$ is $5/3$. That also seems correct. $f(0)$: $3 * 0$ is 0, minus 2 is -2. Denominator: $0 - 2$ is -2. So, $-2/-2$ is 1. That's correct too. Adding them: $2 + 5/3 + 1$. Let me convert 2 and 1 to thirds: 2 is $6/3$, 1 is $3/3$. So, $6/3 + 5/3 + 3/3$ is indeed $14/3$. Hmm, $14/3$ is equal to 4 and $2/3$, which is approximately 4.666...
Wait, but let me think again. Is there a possibility that I made a mistake in substituting the values? Let me check each substitution again. For $f(-2)$: $(3 * (-2) - 2)/(-2 - 2) = (-6 - 2)/(-4) = (-8)/(-4) = 2$. Correct. For $f(-1)$: $(3 * (-1) - 2)/(-1 - 2) = (-3 - 2)/(-3) = (-5)/(-3) = 5/3$. Correct. For $f(0)$: $(3 * 0 - 2)/(0 - 2) = (-2)/(-2) = 1$. Correct. So, adding them: $2 + 5/3 + 1$. Let me add 2 and 1 first: $2 + 1$ is 3. Then, $3 + 5/3$. 3 is $9/3$, so $9/3 + 5/3$ is $14/3$. So, $14/3$ is the correct sum.
Wait, but $14/3$ is an improper fraction, and the question says to express the answer as a common fraction. So, $14/3$ is already a common fraction, so that should be the answer. But just to make sure, let me think if there's another way to approach this problem, maybe simplifying the function first before plugging in the values. Sometimes, simplifying can make the calculations easier or reveal any potential issues. Looking at $f(x) = (3x - 2)/(x - 2)$. Hmm, can I factor anything here? Let me see. The numerator is $3x - 2$, which doesn't factor nicely with the denominator $x - 2$. So, I don't think there's a way to simplify this function further. So, I think my initial approach is correct. Alternatively, maybe I can perform polynomial division or something, but since it's a linear over linear function, it's just a straight division.
Wait, another thought: maybe I can write $f(x)$ as $3 +$ something. Let me try that. Let me rewrite $f(x)$ as $(3x-2)/(x-2)$. Let me perform the division: Divide $3x-2$ by $x-2$. So, how many times does $x-2$ go into $3x-2$? Well, x goes into $3x$ three times. So, $3*(x-2)$ is $3x-6$. Subtract that from $3x-2$: $(3x-2)-(3x-6) = 0x+4$. So, the division gives 3 with a remainder of 4. Therefore, $f(x)$ can be written as $3+4/(x-2)$. So, $f(x) = 3+4/(x-2)$. Hmm, that might be helpful for checking the values. Let me compute $f(-2)$ using this expression: $3+4/(-2-2) = 3+4/(-4) = 3-1 = 2$. Correct, same as before. $f(-1)$: $3+4/(-1-2) = 3+4/(-3) = 3-4/3 = 5/3$. Correct. $f(0)$: $3+4/(0-2) = 3+4/(-2) = 3-2 = 1$. Correct. So, that's another way to confirm the values. So, adding them together: $2 + 5/3 + 1 = 14/3$. **Wait** but just to make sure, let me compute each step again: $f(-2)$: $3 * (-2) = -6$, $-6 - 2 = -8$. Denominator: $-2 - 2 = -4$. So, $-8/-4 = 2$. Correct. $f(-1)$: $3 * (-1) = -3$, $-3 - 2 = -5$. Denominator: $-1 - 2 = -3$. So, $-5/-3 = 5/3$. Correct. $f(0)$: $3 * 0 = 0$, $0 - 2 = -2$. Denominator: $0 - 2 = -2$. So, $-2/-2 = 1$. Correct. Adding them: $2 + 5/3 + 1$. Let me convert 2 and 1 to thirds: 2 is $6/3$, 1 is $3/3$. So, $6/3 + 5/3 + 3/3 = (6 + 5 + 3)/3 = 14/3$. So, $14/3$ is the sum.
Wait, $14/3$ is an improper fraction, but the question says to express the answer as a common fraction. $14/3$ is already a common fraction, so that's fine. **Alternatively**, if I wanted to write it as a mixed number, it would be 4 and $2/3$, but since the question specifies a common fraction, $14/3$ is the appropriate answer. Just to make sure I didn't make any arithmetic errors, let me add $2 + 5/3 + 1$ again: $2 + 1$ is 3, and $3 + 5/3$ is $3 + 1$ and $2/3$, which is 4 and $2/3$, which is $14/3$. Yep, that's correct. So, I think I've double-checked all my steps, and everything seems consistent. Therefore, the value of $f(-2) + f(-1) + f(0)$ is $14/3$.

Final Answer The value of $f(-2) + f(-1) + f(0)$ is $\boxed{\frac{14}{3}}$.

To find the value of $f(-2) + f(-1) + f(0)$ for the function $f(x) = \frac{3x-2}{x-2}$, we need to evaluate the function at each of these points and then sum the results. First, we compute $f(-2)$: $f(-2) = \frac{3(-2)-2}{-2-2} = \frac{-6-2}{-4} = \frac{-8}{-4} = 2$ Next, we compute $f(-1)$: $f(-1) = \frac{3(-1)-2}{-1-2} = \frac{-3-2}{-3} = \frac{-5}{-3} = \frac{5}{3}$ Then, we compute $f(0)$: $f(0) = \frac{3(0)-2}{0-2} = \frac{0-2}{-2} = \frac{-2}{-2} = 1$ Adding these values together: $f(-2) + f(-1) + f(0) = 2 + \frac{5}{3} + 1$ Converting 2 and 1 to fractions with a common denominator: $2 = \frac{6}{3}$ and $1 = \frac{3}{3}$ Adding the fractions: $\frac{6}{3} + \frac{5}{3} + \frac{3}{3} = \frac{6+5+3}{3} = \frac{14}{3}$ Thus, the value of $f(-2) + f(-1) + f(0)$ is $\boxed{\frac{14}{3}}$

Figure 9: Inference case. The left side is the original DeepSeek-R1-Distill-Qwen-1.5B model, and the right side is our MuTIS model.