Breaking Contextual Inertia: Reinforcement Learning with Single-Turn Anchors for Stable Multi-Turn Interaction

Xingwu Chen[1], Zhanqiu Zhang[2,†], Yiwen Guo[3,†], Difan Zou[1,†]
School of Computing and Data Science, The University of Hong Kong [1]
LIGHTSPEED[2]
Independent Researcher[3]
xingwu@connect.hku.hk, zqzhang27@gmail.com,
guoyiwen89@gmail.com, dzou@cs.hku.hk

Abstract

While LLMs demonstrate strong reasoning capabilities when provided with full information in a single turn, they exhibit substantial vulnerability in multi-turn interactions. Specifically, when information is revealed incrementally or requires updates, models frequently fail to integrate new constraints, leading to a collapse in performance compared to their single-turn baselines. We term the root cause as *Contextual Inertia*: a phenomenon where models rigidly adhere to previous reasoning traces. Even when users explicitly provide corrections or new data in later turns, the model ignores them, preferring to maintain consistency with its previous (incorrect) reasoning path. To address this, we introduce **R**einforcement **L**earning with **S**ingle-**T**urn **A**nchors (**RLSTA**), a generalizable training approach designed to stabilize multi-turn interaction across diverse scenarios and domains. RLSTA leverages the model's superior single-turn capabilities as stable internal anchors to provide reward signals. By aligning multi-turn responses with these anchors, RLSTA empowers models to break contextual inertia and self-calibrate their reasoning based on the latest information. Experiments show that RLSTA significantly outperforms standard fine-tuning and abstention-based methods. Notably, our method exhibits strong cross-domain generalization (e.g., math to code) and proves effective even without external verifiers, highlighting its potential for general-domain applications.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in solving complex reasoning tasks (Yang et al., 2025; Google, 2025). However, their success has predominantly been confined to single-turn settings (Yang et al., 2025) or scenarios involving carefully engineered inference procedures (Shao et al., 2025). Yet, multi-turn interaction has emerged as the ubiquitous paradigm for human-AI engagement, serving as the backbone for applications ranging from general-purpose chat systems (OpenAI, 2025a; Google, 2025) to complex agentic workflows (Liu et al., 2025; Wei et al., 2025). In these practical scenarios, users frequently introduce new conditions or correct previous requirements. However, models frequently fail to refresh their reasoning in multi-turn contexts, yielding unsatisfactory answers compared to single-turn scenarios. For instance, as illustrated in Figure 1, when the user updates the budget constraint, the model rigidly adheres to its previous "ride-hailing" plan and suggests spending time finding a carpool, producing a solution that directly violates the urgency requirement. This deficiency severely limits LLM utility in complex applications like interactive reasoning (Wu et al., 2025) and collaborative problem-solving (Chen et al., 2024).

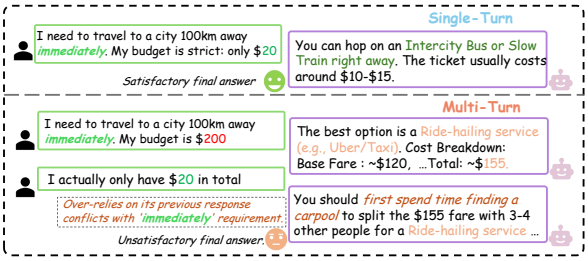

Figure 1: Contextual Inertia in multi-turn interaction: The persistence of the initial response leads to an unsatisfactory final answer.

Laban et al. (2025) refer to this performance degradation in multi-turn settings as getting "Lost in Conversation" (LiC). They focus on a specific scenario where a complete condition is fragmented into multiple shards and provided to the LLM sequentially (`MT-Add`). Through large-scale experiments, they reveal that models are vulnerable in such scenarios, as LLMs often generate premature answer attempts in early turns without full information. These premature attempts can adversely affect follow-up responses, ultimately causing the model to lose track of the logical flow.

[†] Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2026, pages 6302–6318
July 2-7, 2026

Existing approaches primarily address the symptoms of multi-turn degradation rather than investigating its root cause. One prominent line of research seeks to enhance multi-turn capabilities through direct fine-tuning strategies (Sun et al., 2024; Zhou et al., 2024, 2025). While effective for general instruction following and credit assignment, these methods rely heavily on external supervision, thereby bypassing the intrinsic degradation mechanisms rather than rectifying the model's internal failure modes. Another direction investigates behavioral strategies, such as prompting clarification requests (Wu et al., 2025) or encouraging active abstention (Li, 2025), particularly when user input is insufficient. Although these methods mitigate error accumulation from premature responses, they are incompatible with scenarios necessitating dynamic state updates (Kwan et al., 2024; Bai et al., 2024), such as the `MT-Refine` setting, where the model must iteratively correct its initial response rather than remaining silent.

In this paper, we identify the **indiscriminate nature** of the model's multi-turn behavior and **quantitatively attribute** the root cause of multi-turn interaction failures to a phenomenon we term *Contextual Inertia*. Specifically, LLMs in multi-turn dialogues tend to rigidly adhere to previously generated reasoning traces, even when these traces have been explicitly negated or corrected by subsequent user information. This indiscriminate nature causes erroneous or misleading intermediate reasoning to be continuously inherited and reinforced throughout the multi-turn iteration, ultimately leading to incorrect final answers. Our analysis reveals that over 70%–90% of multi-turn errors can be directly traced back to the propagation of errors from previous turn responses, rather than independent reasoning failures in the final turn. Therefore, breaking such inertia and pushing the model to correct its previous failures serves as a critical and general step toward stable multi-turn interaction.

Building on this insight, we introduce **R**einforcement **L**earning with **S**ingle-**T**urn **A**nchors (**RLSTA**), a generalizable training approach designed to break contextual inertia and stabilize multi-turn interaction. RLSTA capitalizes on the model's superior reasoning capabilities in full-information single-turn settings, treating these responses as stable internal *anchors* to guide multi-turn generation via reward signals. Our contributions can be summarized as follows:

- We identify the **indiscriminate nature** of *Contextual Inertia* and **quantitatively attribute** its impact. We define this phenomenon as the model's indiscriminate adherence to previous reasoning traces, providing a new perspective on addressing failures in dynamic interactions.
- We propose Reinforcement Learning with Single-Turn Anchors (RLSTA), a generalizable training approach designed to break *Contextual Inertia* by leveraging the model's superior single-turn capabilities as internal reward signals. Unlike previous methods tailored for specific tasks, RLSTA is applicable to diverse interaction scenarios, including both incremental information addition (`MT-Add`) and error correction (`MT-Refine`).
- We conduct extensive experiments demonstrating that RLSTA not only outperforms standard tuning methods but also achieves comparable or even superior performance compared to abstention-based fine-tuning approaches. Moreover, RLSTA exhibits cross-domain generalization and proves effective even without external verifiers, highlighting its potential for general-domain applications.

2 Related Works

LLM's Vulnerability in Multi-Turn Interaction Large Language Models (LLMs) have demonstrated significant struggles in multi-turn tasks, spanning human-model interaction (Laban et al., 2025; Bai et al., 2024; Kwan et al., 2024), agentic inference (Zhou et al., 2025), and tool use (Patil et al., 2025; Wang et al., 2023). Specifically, Kwan et al. (2024) introduced MT-Eval and observed an average 15% performance degradation compared to single-turn settings. Laban et al. (2025) further introduced a more challenging multi-turn setting involving underspecified requirements revealed sequentially, identifying a severe 39% performance drop, they term this degradation "Lost-in-Conversation" (LiC) and attribute the root cause to premature answering attempts in early turns. In this work, we investigate both the `MT-Add` scenario, where information is added incrementally, and the `MT-Refine` scenario, where the user corrects initially incorrect conditions. Where LLM exhibits substantial vulnerability.

Optimization for Multi-Turn Interaction To enhance model performance in multi-turn settings, a line of work attempts to directly implement various fine-tuning methods, including Supervised Fine-Tuning (SFT) (Chiang et al., 2023; Ding et al.,

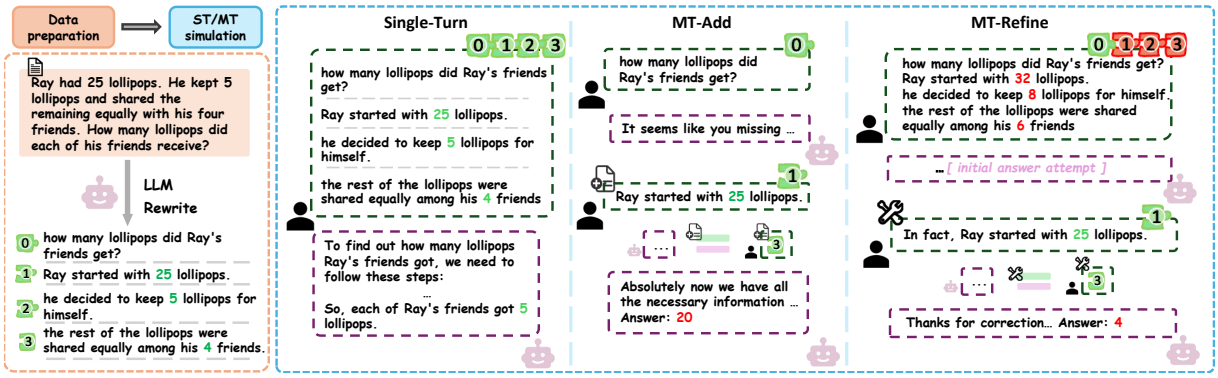

Figure 2: Overview of our data preparation and multi-turn simulation pipeline. We partition single-turn prompts into segments to simulate two multi-turn scenarios: `MT-Add` (incremental information addition) and `MT-Refine` (correction of initially incorrect conditions).

2023; Sun et al., 2024), Direct Preference Optimization (Shi et al., 2024; Xiong et al., 2024), and Reinforcement Learning (Zhou et al., 2024; Wang et al., 2025; Kumar et al., 2024). While effective for general instruction following, these approaches bypass the intrinsic mechanism of degradation by relying on external guidance rather than correcting the model's internal failure modes. Moreover, methods like MT-PPO (Zeng et al., 2025a) and SWEET-RL (Zhou et al., 2025) require expensive, well-designed turn-level rewards, which are unsuitable for general practical settings. Another direction explores behavioral strategies for specific scenarios, such as encouraging clarification requests (Wu et al., 2025) or active abstention (Li, 2025) when the information provided by users is insufficient. These methods aim to mitigate error accumulation from premature responses; however, such strategies are incompatible with scenarios requiring state updates, such as the `MT-Refine` setting addressed in this paper. In these cases, the model must generate an initial response and subsequently correct it, rather than simply remaining silent.

3 Problem Setup

Most existing benchmarks evaluate models using a single-turn prompt (i^{single}), ignoring the interactive nature of practical LLM usage. In this work, we evaluate LLMs solving problems in a multi-turn setting, where users provide information $\{i_0, i_1, \ldots, i_n\}$ sequentially across turns. Specifically, we consider two scenarios: `MT-Add`, where information is added incrementally, and `MT-Refine`, where the user corrects initially incorrect conditions, as shown in Figure 2.

`MT-Add`. In this scenario, we partition the original single-turn prompt i^{single} into multiple segments $\{i_0, i_1, \ldots, i_n\}$. Here, i_0 represents the initial prompt specifying the problem target (e.g., "What is Ara's total score?"), while subsequent prompts provide the additional constraints necessary to solve the problem. We assume an interaction paradigm where the user incrementally provides new information in each turn. This mirrors practical scenarios where users consistently provide follow-up information to guide the LLM toward a satisfactory answer.

`MT-Refine`. Here, we simulate error correction. The user first provides a prompt with full but corrupted information $i_0 = i^{\text{corrupt}}$, where key conditions from the ground truth conditions in $\{i_1, \ldots, i_n\}$ are replaced with incorrect values. The user then corrects these specific errors sequentially through subsequent turns.

During the multi-turn conversation, information is presented sequentially. At the k-th turn, we use i_k as the user's new query and provide the model with the conversation history $H = \{s, i_0, m_0, \ldots, i_{k-1}, m_{k-1}, i_k\}$, where s is the system prompt and m_j denotes the model's response at turn j. We take the final-turn response m_n as the final multi-turn answer. For comparison, we evaluate the model's single-turn performance by providing the full correct information $i^{\text{full}} = \texttt{merge}(i_0, \ldots, i_n)$ in a single turn, taking the resulting response m as the final single-turn answer.

4 Contextual Inertia: Characterization and Quantitative Analysis

While recent studies have identified that LLMs struggle in multi-turn interactions (Laban et al., 2025; Gupta et al., 2024; Hankache et al., 2025), the specific drivers of this degradation remain under-explored. Existing works primarily attribute

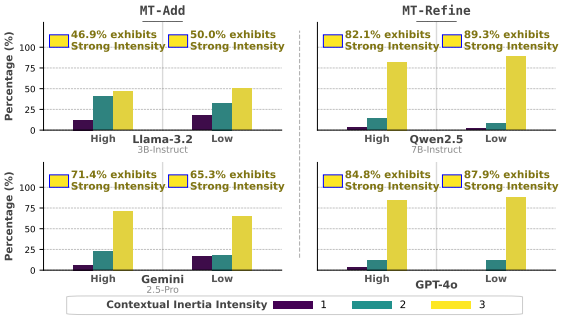

Figure 3: Distributions of Contextual Inertia Intensity $\mathtt{I}_{\mathrm{CI}}(m_n, m_{n-1})$. We use GPT-4o to categorize the inertia intensity as Weak (1), Moderate (2), and Strong (3). In most cases, the model's final answer m_n exhibits strong inertia intensity to the preceding response m_{n-1}. Notably, the distribution of this intensity remains indistinguishable regardless of the conversation history quality (high vs. low), providing empirical evidence for the indiscriminate nature of contextual inertia.

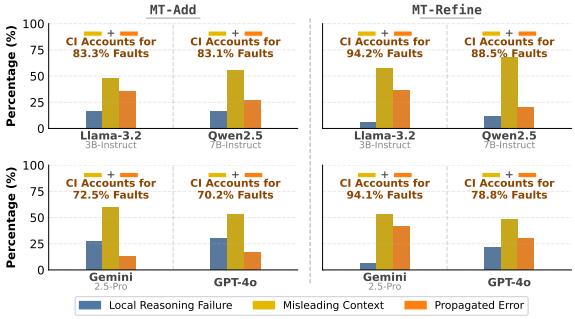

Figure 4: Root cause of failures in multi-turn conversations: failures predominantly originate from previous responses (**Misleading Context** and **Propagated Error**), which are driven by Contextual Inertia.

failures to context length or premature answering (Laban et al., 2025). In this section, we advance this understanding by analyzing the **indiscriminate nature** of the model's persistence and providing a **quantitative attribution** of these failures. We term this specific behavioral pattern as **Contextual Inertia**: *the inherent tendency of LLMs to rigidly adhere to previous reasoning traces when processing new instructions, even when those traces are partially invalid or obsolete.*

Indiscriminate Nature of Contextual Inertia. We first qualify and statistically analyze the contextual inertia. Formally, let $H = \{s, i_0, m_0, \ldots, m_{n-1}, i_n\}$ denote a conversation history, we define the intensity of contextual inertia $\mathtt{I}_{\mathrm{CI}}(m_n, m_{n-1})$ as the semantic similarity between the model's final answer m_n and the preceding response m_{n-1}. We categorize histories into two sets: *high-quality* histories $\mathcal{H}_{\text{high}}$, which result in a correct final answer $m_n \sim \pi(\cdot|H)$, and *low-quality* histories $\mathcal{H}_{\text{low}}$, which lead to an incorrect answer. We reveal the indiscriminate nature of contextual inertia, where the intensity distributions conditioned on contrasting history qualities are statistically indistinguishable. That is:

$$\begin{aligned} &\mathbb{P}\left(\mathtt{I}_{\mathrm{CI}}(m_n, m_{n-1}) \mid H \in \mathcal{H}_{\text{high}}\right) \\ \approx\ &\mathbb{P}\left(\mathtt{I}_{\mathrm{CI}}(m_n, m_{n-1}) \mid H \in \mathcal{H}_{\text{low}}\right). \end{aligned} \tag{1}$$

To empirically validate this, we employ GPT-4o (Hurst et al., 2024) to analyze the semantic similarity (e.g., logical structure, chain-of-thought steps) between the model's final answer m_n and the preceding response m_{n-1}. As shown in Figure 3, we observe that the intensity distributions are remarkably similar across both groups. Moreover, the model's final answer m_n and the last response m_{n-1} exhibit high similarity in most cases. This validates the indiscriminate nature of contextual inertia: even when m_{n-1} is misleading or erroneous, the model is compelled to propagate these traces into subsequent turns.

Quantifying the Dominance of Contextual Inertia. Beyond statistical similarity, we further trace the sources of error by collecting multi-turn conversations $\{s, i_0, \ldots, i_n, m_n\}$ where the final answer m_n is verified as incorrect, and prompt GPT-4o to classify the root cause into three categories:[1]

- **Misleading Context:** The failure stems from previous responses $\{m_0, \ldots, m_{n-1}\}$ which, while ostensibly correct, create a misleading or contradictory context for follow-up information.
- **Propagated Error:** The failure is caused by previous responses that are fundamentally wrong, where the model passively inherits and propagates these factual or logical errors to the final response.
- **Local Reasoning Failure:** Previous responses are of generally good quality; the failure occurs independently in the final turn when the model processes the user's latest query i_n (e.g., losing the problem target or making arithmetic errors).

As demonstrated in Figure 4, we find that over 70%–90% of multi-turn errors can be directly traced back to previous responses (**Misleading Context** or **Propagated Error**). In these cases, due to the indiscriminate nature, contextual inertia compels the LLM to adopt misleading or erroneous

[1]To isolate the limitations of the base model, we select conversations where providing full information $i^{\text{full}} = \text{merge}(i_0, \ldots, i_n)$ in a single turn yields high (> 0.7) accuracy.

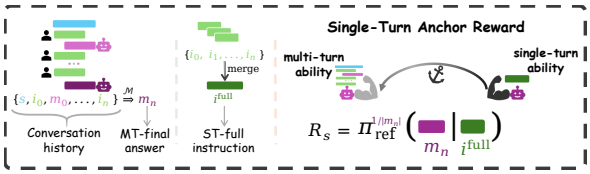

Figure 5: **Single-Turn Anchor Reward** (R_s)**.** We leverage the model's superior single-turn ability on full instruction (i^{full}) as anchor for the multi-turn final answer m_n. Our filtering strategy (Equation 2) ensures the anchor is reliable by retaining only histories where single-turn performance exceeds multi-turn.

traces without verification, causing error accumulation across the conversation that ultimately degrades performance.

Due to its indiscriminate nature, Contextual Inertia serves as the primary driver of model vulnerability in multi-turn settings. While recent studies (Wan et al.; Laban et al., 2025; Castillo-Bolado et al., 2024) demonstrate that LLMs tend to attend heavily to the last response, we go a step further by revealing the indiscriminate nature of this phenomenon in multi-turn conversations. This insight motivates our approach to break this inertia for stable interaction. We detail our methodology in Section 5.

5 Approach

To address *Contextual Inertia* in multi-turn LLMs, where models rigidly adhere to obsolete reasoning traces, we propose **R**einforcement **L**earning with **S**ingle-**T**urn **A**nchors (**RLSTA**).

Existing solutions typically attribute multi-turn vulnerability to "uncertainty", mitigating it through abstention (Li, 2025) or clarification requests (Wu et al., 2025) in early turns to avoid generating premature answers based on insufficient information. While viable for handling ambiguity in `MT-Add`, these strategies merely address the symptoms rather than curing the root cause: Contextual Inertia. Furthermore, such approaches are incompatible with scenarios like `MT-Refine`, where the interaction relies on the model providing an initial response that the user subsequently corrects (e.g., *"No, actually X is 5"*). In these cases, the model cannot abstain in early turns; it must fundamentally overcome its inertia for a correct answer.

We target this root cause directly. Instead of resorting to passive abstention, we aim to *calibrate* the model's multi-turn reasoning against its own superior single-turn capabilities. Our approach combines: (i) Latent Capability Filtering, which isolates cases where the model successfully solves the problem with the full instruction (i^{full}), yet fails to generate the correct answer under the sequential multi-turn history (H) due to contextual inertia. (ii) An anchor-based RL method that utilizes the model's single-turn reasoning ability as internal anchors. We detail our approach as follows.

5.1 Latent Capability Filtering

In Section 4, we identify contextual inertia within potentially misleading or erroneous conversation histories as the primary driver of multi-turn vulnerability in LLMs. Building on this insight, we aim to rectify such behavior by specifically optimizing the last-turn response m_n in cases where the multi-turn conversation history $H = \{s, i_0, m_0, \ldots, i_{n-1}, m_{n-1}, i_n\}$ leads to performance degradation.

We first collect a raw set of multi-turn conversation histories $\mathcal{D}_{\text{raw}}$. We then perform **Latent Capability Filtering** to ensure we can use the model's single-turn ability as a stable anchor for the model's final response given multi-turn conversation history. Specifically, we isolate instances where the model possesses the latent capability to solve the problem given the full information $i^{\text{full}} = \texttt{merge}(i_0, \cdots, i_{n-1}, i_n)$, yet fails to do so under the original multi-turn history H. Formally, we retain conversation histories satisfying the following condition:

$$\mathbb{E}_{m\sim\pi(\cdot|i^{\text{full}})}[\texttt{Ver}(m)] > \mathbb{E}_{m_n\sim\pi(\cdot|H)}[\texttt{Ver}(m_n)]. \quad (2)$$

Where $\texttt{Ver}(\cdot)$ is the verifier to check the correctness of the final answer with $0/1$, this filtering process yields the dataset $\mathcal{D}_{\mathcal{M}}$ for model $\mathcal{M}$. Our objective is to break contextual inertia in scenarios where the preceding conversation is low quality. Therefore, we exclude conversation histories H where single-turn and multi-turn performances are comparable. This exclusion allows us to focus on instances where the model possesses superior single-turn capabilities, thereby providing a high-quality supervision signal for alignment, which we formulate as our single-turn anchor for RL training.

5.2 RL with Single-Turn Anchors

We use GRPO as our training algorithm. For a given conversation history H, GRPO samples a group of outputs $\{m_n^{(1)}, \ldots, m_n^{(G)}\}$ from the old policy θ_{old} and optimizes the policy model by max-

imizing:

$$\mathcal{J}_{\text{GRPO}}(\theta)=\mathbb{E}\Big[H\sim\mathcal{D}_{\mathcal{M}},\{m_n^{(i)}\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|H)\Big]$$
$$\sum_{i=1}^{G}\frac{1}{|m_n^{(i)}|}\sum_{t=1}^{|m_n^{(i)}|}\Big\{\min\Big[r_{i,t}(\theta)\hat{A}_{i,t},\ \text{clip}(r_{i,t}(\theta),1-\epsilon,1+\epsilon)\hat{A}_{i,t}\Big]$$
$$-\beta\mathbb{D}_{\text{KL}}[\pi_\theta||\pi_{\text{ref}}]\Big\}, \tag{3}$$

where

$$r_{i,t}(\theta)=\frac{\pi_\theta\left(m_{n,t}^{(i)}|H,m_{n,<t}^{(i)}\right)}{\pi_{\theta_{\text{old}}}\left(m_{n,t}^{(i)}|H,m_{n,<t}^{(i)}\right)},\quad \hat{A}_{i,t}=\frac{R_i-\text{mean}\left(\{R_i\}_{i=1}^{G}\right)}{\text{std}\left(\{R_i\}_{i=1}^{G}\right)}.$$

Here, R_i denotes the reward for response $m_n^{(i)}$, which is typically a rule-based reward, such as using a verifier to check the final answer or to check whether the response follows a required format (Guo et al., 2025; Zeng et al., 2025b).

In addition to the outcome-based verification reward $R_{\text{v}} = \texttt{Ver}(m_n^{(i)})$, we introduce a mechanism to stabilize the reasoning process: the **Single-Turn Anchor Reward** (R_{s}). This component utilizes the model's strong ability under the full-information single-turn setting (i^{full}) as a stable anchor for the current multi-turn generation $m_n^{(i)}$:

$$R_{\text{s}}=\left(\prod_{t=1}^{|m_n^{(i)}|}\pi_{\theta_{\text{ref}}}\left(m_{n,t}^{(i)}\mid i^{\text{full}},m_{n,<t}^{(i)}\right)\right)^{\frac{1}{|m_n^{(i)}|}}, \tag{4}$$

Here, i^{full} merges all user conditions from the multi-turn history H into a single-turn query, and $\pi_{\theta_{\text{ref}}}$ denotes the base model. By calculating the length-normalized likelihood of the response under $\pi_{\theta_{\text{ref}}}(\cdot \mid i^{\text{full}})$, R_{s} quantifies how well the multi-turn response aligns with the model's superior single-turn capabilities.

Combining the outcome reward R_{v} and the anchor reward R_{s} yields the final reward for response $m_n^{(i)}$:

$$R = R_{\text{v}} + \alpha R_{\text{s}}, \tag{5}$$

where α is a hyperparameter. We set $\alpha = 1/G$ so that the advantage $\hat{A}$ for a correct response with $R_{\text{v}} = 1$ remains positive.

We highlight that R_{s} serves as a critical behavioral **anchor**. By leveraging our data-filtering strategy (Equation 2), which ensures that the single-turn policy $\pi_{\theta_{\text{ref}}}(\cdot \mid i^{\text{full}})$ demonstrates superior accuracy compared to the noisy multi-turn history $\pi_\theta(\cdot \mid H)$, R_{s} provides a robust supervision signal. This signal effectively pulls the generation process away from the bias of contextual inertia, anchoring the model to the correct reasoning path established in the single-turn setting.

6 Experiments

We evaluate our method across various multi-turn scenarios, tasks, and training configurations in controlled settings. The results demonstrate that our method effectively aligns the model's multi-turn performance, yielding superior results and promising generalization to other tasks, while preserving the base model's single-turn performance and long-context capabilities. We detail our experimental setup in Subsection 6.1, followed by the main results and comparisons with existing training methods and other models in Subsection 6.2. Finally, we examine the preservation of long-context capabilities and analyze the training dynamics of the single-turn anchor reward in Subsection 6.3.

6.1 Experiments Setup

Here we introduce the tasks, datasets, and models used in our experiments. Additional details are provided in Appendix B.

Data Preparation. We construct our dataset following the instruction segmentation protocol proposed by Laban et al. (2025). For training, we derive multi-turn samples from the GSM8K dataset (Cobbe et al., 2021) by employing GPT-4o (Hurst et al., 2024) to decompose original single-turn queries into sequential instructions. For evaluation, we adopt the multi-turn benchmarks from Laban et al. (2025). Notably, while our model is trained exclusively on math-domain multi-turn scenarios, we evaluate it on diverse domains, including code and summarization, to assess its cross-domain generalization capabilities.

Multi-turn Scenario Simulation. We simulate two distinct interaction patterns: `MT-Add` and `MT-Refine`. In the `MT-Add` setting, segmented instructions $\{i_0, i_1, \cdots, i_n\}$ are presented sequentially, requiring the model to integrate new constraints incrementally. The `MT-Refine` setting simulates error correction. Here, we employ GPT-4o to alter key conditions within the segments $\{i_1, \cdots, i_n\}$ and concatenate these modified conditions with the initial prompt i_0. This forms a corrupted initial instruction i^{corrupt}, while the original valid segments are provided in subsequent turns as user corrections. Note that the code and summary tasks are evaluated solely under the `MT-Add` setting due to data compatibility. Beyond fixed datasets, following Laban et al. (2025), we also assess performance in a dynamic setting using GPT-4o-mini

Table 1: Experimental results on MT-Add and MT-Refine Tasks. We compare RLSTA with SFT, DPO, and vanilla GRPO. **Bold** indicates best performance, underline indicates improvement over Base, and gray indicates performance lower than Base.

Method	Qwen2.5-3B-Instruct				Qwen2.5-7B-Instruct				Qwen3-4B-Instruct-2507				Llama-3.2-3B-Instruct			
	MT-Add		MT-Refine	Avg	MT-Add		MT-Refine	Avg	MT-Add		MT-Refine	Avg	MT-Add		MT-Refine	Avg
	Math	Code	Math		Math	Code	Math		Math	Code	Math		Math	Code	Math	
Base	0.493	0.220	0.603	0.439	0.638	0.312	0.669	0.540	0.772	0.370	0.814	0.652	0.470	0.140	0.585	0.398
SFT	0.546	0.203	0.533	0.427	0.740	0.310	0.694	0.581	0.826	0.444	0.809	0.693	0.568	0.177	0.519	0.421
DPO	0.546	0.203	0.568	0.439	0.633	0.312	0.522	0.489	0.784	0.416	0.801	0.667	0.559	0.198	0.368	0.375
GRPO	0.636	0.190	0.734	0.520	0.803	0.336	**0.836**	0.659	0.839	0.472	0.882	0.731	0.608	0.190	0.620	0.473
RLSTA (Ours)	**0.715**	**0.256**	**0.745**	**0.572**	**0.857**	**0.350**	0.822	**0.676**	**0.903**	**0.552**	**0.898**	**0.784**	**0.649**	**0.205**	**0.640**	**0.498**

as a user simulator. This simulator interactively selects instructions rather than following a fixed sequence, thereby more closely mimicking realistic human interactions.

Models and Baselines. We validate our method across a range of open-weights models, including Qwen2.5-3B/7B-Instruct (Yang et al., 2024), Qwen3-4B-2507 (Yang et al., 2025), and Llama-3.2-3B-Instruct (Grattafiori et al., 2024). We employ GRPO as our primary training algorithm and compare RLSTA against standard fine-tuning baselines: Supervised Fine-Tuning (SFT), Direct Preference Optimization (DPO) (Rafailov et al., 2023), and vanilla GRPO. For SFT and DPO, we construct training pairs by appending the correct single-turn response to the multi-turn history, and utilize the model's original erroneous response as the negative sample for DPO. Additionally, we benchmark against uncertainty-handling strategies: RLAAR (Li, 2025) and CollabLLM (Wu et al., 2025).

6.2 Main Results

RLSTA Can Effectively Break Contextual Inertia. We begin by analyzing the distributions of contextual inertia intensity after RLSTA on MT-Add. Following the methodology in Section 4, we categorize conversation histories based on the base model's performance: "High" quality histories (leading to correct answers) and "Low" quality ones (leading to incorrect answers). While the base model exhibits indiscriminate inertia regardless of the quality of conversation history (Figure 3), RLSTA maintains high similarity for the High-quality group but shows notably lower contextual inertia intensity for the Low-quality group (Figure 6). This confirms that RLSTA successfully break the indiscriminate nature of contextual inertia, while preserving the ability to utilize beneficial history.

RLSTA Outperforms Standard Fine-Tuning Methods. As shown in Table 1, RLSTA significantly improves the model's performance across

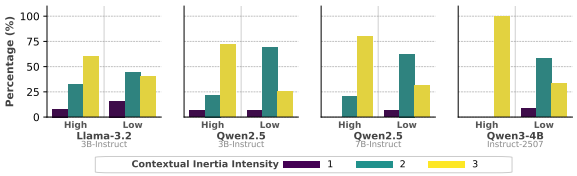

Figure 6: The distributions of contextual inertia intensity after RLSTA. Compared with base model in Figure 3, RLSTA successfully breaks the contextual inertia, while preserving the ability to utilize beneficial history.

different multi-turn scenarios compared to SFT, DPO, and GRPO. Notably, although we only train the model using math-domain multi-turn scenarios, RLSTA demonstrates strong generalization to other domains (e.g., Code). This suggests that RLSTA fosters a fundamental resilience against contextual inertia on misleading conversation history, enabling the model to maintain robust reasoning capabilities even in domains outside its training distribution.

Table 2: Comparison of LiC Scores (↑) between RLSTA and RLAAR on MT-Add scenario. Underline indicates improvement over the Base Model.

Method	Math	Actions	Database	Code	Avg
Base	0.728	0.365	0.564	0.525	0.546
RLAAR	0.950	0.530	0.873	0.702	0.764
RLSTA	1.001	0.688	0.691	0.596	0.744
RLSTA (+ Abstain)	0.988	0.702	0.762	0.642	0.773

Table 3: Comparison between RLSTA and CollabLLM on MT-Add scenario, we report model performance (↑) and token consumption (↓) (Performance/Tokens)

Method	Code		Math	
	Sing-turn	Multi-turn	Sing-turn	Multi-turn
Base	0.320/142.3	0.223/360.2	0.782/224.8	0.443/319.9
CollabLLM	0.302/125.2	0.214/**321.5**	0.758/250.0	0.433/**294.6**
RLSTA	0.343/149.4	**0.309**/339.3	0.754/230.4	**0.654**/374.0

RLSTA Achieves Comparable Performance to Abstention and Active Inquiry Methods. We further benchmark our method against uncertainty-handling strategies: RLAAR (Li, 2025), which encourages abstention under uncertainty, and CollabLLM (Wu et al., 2025), which tunes the model to actively request more information. For RLAAR, we reference the results reported in (Li, 2025) us-

ing Qwen2.5-7B-Instruct as the base model. As presented in Table 2, RLSTA achieves an average LiC Score (the ratio between multi and single turn performance) comparable to RLAAR, while significantly outperforming it on the *Math* and *Actions* tasks. Although RLAAR maintains a marginal advantage in Code and Database by leveraging code training data, RLSTA achieves competitive results without such domain-specific supervision, demonstrating strong cross-domain generalization capabilities. Furthermore, when explicitly instructing the model to abstain via the system prompt (RLSTA + Abstain), our method yields higher average performance than RLAAR. Regarding CollabLLM, we employ an LLM-based user simulator to create realistic multi-turn scenarios, evaluating both performance and token efficiency using Llama-3.1-8B-Instruct as the base model. As detailed in Table 3, although CollabLLM reduces token consumption, it fails to deliver performance improvements on our `MT-Add` tasks under this dynamic setting. In contrast, RLSTA achieves substantial gains (36.9% on code, 47.6% on math) with competitive efficiency (utilizing 5.8% fewer tokens for code with only a moderate 17.2% increase for math). Crucially, unlike these baselines, RLSTA improves multi-turn performance without relying on passive abstention, thereby serving as a generalizable approach to a wider range of scenarios, like `MT-Refine`, to which uncertainty-handling strategies are inapplicable.

6.3 Long-Context Preservation and Training Efficiency

Table 4: Model performance (Coverage Score ↑) on the Summary Task across multi and single turn scenarios. RLSTA maintains, and in most cases improves, the model's long-context capabilities while simultaneously stabilizing multi-turn interactions.

Model	Method	Multi-turn	Single-turn	LiC Score
Qwen2.5-3B Instruct	Base	0.359	0.524	0.685
	RLSTA	0.433	0.520	**0.832**
Qwen2.5-7B Instruct	Base	0.446	0.594	0.751
	RLSTA	0.441	0.590	0.747
Qwen3-4B Instruct-2507	Base	0.555	0.750	0.740
	RLSTA	0.632	0.758	**0.835**
Llama-3.2-3B Instruct	Base	0.384	0.549	0.698
	RLSTA	0.434	0.555	**0.782**

RLSTA Maintains Long-Context Processing Ability. We further check whether breaking contextual inertia on potentially erroneous history affects the model's ability to process long contexts, we evaluate performance on the multi-turn Summary task. Here, the model is provided with very long contexts (spanning tens of thousands of tokens) across multiple turns and instructed to summarize the previous context in each turn. We use *Coverage Scores* (Laban et al., 2024) as the metric, which evaluates the model's capacity to process long contexts. As shown in Table 4, compared to the Base model, RLSTA maintains or even improves model's long-context processing ability. This demonstrates that RLSTA successfully breaks contextual inertia without compromising the model's ability to utilize conversation history in long-context scenarios.

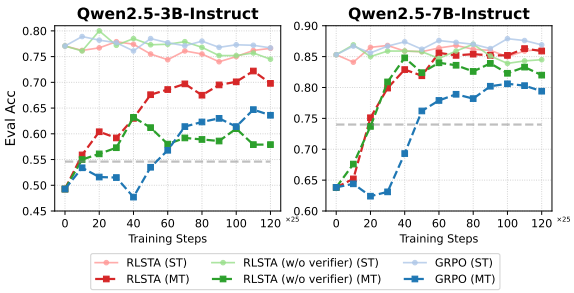

Figure 7: Training dynamics of Single-turn (ST) and Multi-turn (MT) performance across different methods. The gray dashed line (- - -) represents the corresponding SFT MT performance baseline. Notably, even in the absence of external verifiable rewards (**RLSTA w/o verifier**), our method effectively narrows the performance gap between single-turn and multi-turn settings.

Training Dynamics Comparison. Finally, we examine the training dynamics and the efficiency of our single-turn anchor reward R_s. Figure 7 illustrates the trajectory of Single-Turn (ST) and Multi-Turn (MT) performance during training. We observe that compared to standard GRPO, RLSTA exhibits accelerated convergence and superior asymptotic performance.

Crucially, we evaluate a variant, **RLSTA (w/o verifier)**, which relies solely on the internal single-turn anchor reward (R_s) without access to the external ground-truth verifier (R_v).The results demonstrate that even in the absence of external verifiable rewards, RLSTA effectively stabilizes multi-turn interaction and even achieves performance comparable to the full RLSTA setting, highlighting the potential of RLSTA in general domains where external verifiers are unavailable.

7 Conclusion

In this work, we identified the indiscriminate nature of *Contextual Inertia* and quantitatively attributed its impact as the primary cause of LLM vulnerability in multi-turn interaction. We proposed Re-

inforcement Learning with Single-Turn Anchors to address this issue by leveraging the intrinsic single-turn capabilities of the model as stable internal anchors. Extensive experiments demonstrate that our approach significantly enhances stability in both `MT-Add` and `MT-Refine` scenarios while outperforming standard fine-tuning baselines. Moreover, our method exhibits strong cross-domain generalization and data efficiency even in the absence of external verifiers. These findings highlight the potential of our framework to facilitate more reliable and adaptive multi-turn interactions for general applications.

Limitations

The effectiveness of RLSTA is intrinsically bounded by the model's single-turn capabilities, as our method leverages the single-turn response as a supervisory anchor and thus assumes the model possesses the latent knowledge to solve the problem given full information. Additionally, our current framework operates under a passive interaction paradigm where the user voluntarily provides all necessary conditions, leaving scenarios that require proactive clarification unaddressed. Furthermore, while RLSTA offers a universal approach to breaking contextual inertia across various settings like `MT-Add` and `MT-Refine`, it does not yet incorporate meta-cognitive decision-making; given that abstention-based methods can be effective in specific multi-turn scenarios (`MT-Add`), teaching the model to dynamically evaluate context sufficiency and select the optimal strategy, such as reasoning, clarifying, or abstaining, remains a promising direction for future investigation.

References

Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, and Wanli Ouyang. 2024. MT-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7421–7454, Bangkok, Thailand. Association for Computational Linguistics.

David Castillo-Bolado, Joseph Davidson, Finlay Gray, and Marek Rosa. 2024. Beyond prompts: Dynamic conversational benchmarking of large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Pei Chen, Shuai Zhang, and Boran Han. 2024. CoMM: Collaborative multi-agent, multi-reasoning-path prompting for complex problem solving. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1720–1738, Mexico City, Mexico. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, and 1 others. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051.

Google. 2025. A new era of intelligence with gemini 3. Google Blog. Accessed: 2025-11-18.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Akash Gupta, Ivaxi Sheth, Vyas Raina, Mark Gales, and Mario Fritz. 2024. Llm task interference: An initial study on the impact of task-switch in conversational history. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14633–14652.

Robert Hankache, Kingsley Nketia Acheampong, Liang Song, Marek Brynda, Raad Khraishi, and Greig A Cowan. 2025. Evaluating the sensitivity of llms to prior context. *arXiv preprint arXiv:2506.00069*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, and 1 others. 2024. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*.

Wai-Chung Kwan, Xingshan Zeng, Yuxin Jiang, Yufei Wang, Liangyou Li, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. 2024. MT-eval: A multi-turn capabilities evaluation benchmark for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20153–20177, Miami, Florida, USA. Association for Computational Linguistics.

Philippe Laban, Alexander Richard Fabbri, Caiming Xiong, and Chien-Sheng Wu. 2024. Summary of a haystack: A challenge to long-context llms and rag systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9885–9903.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. 2025. Llms get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*.

Ming Li. 2025. Verifiable accuracy and abstention rewards in curriculum rl to alleviate lost-in-conversation. *arXiv preprint arXiv:2510.18731*.

Bang Liu, Xinfeng Li, Jiayi Zhang, Jinlin Wang, Tanjin He, Sirui Hong, Hongzhang Liu, Shaokun Zhang, Kaitao Song, Kunlun Zhu, and 1 others. 2025. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*.

OpenAI. 2025a. Gpt-5 system card. OpenAI Blog. Accessed: 2025-8-7.

OpenAI. 2025b. Openai o3 and o4-mini system card. OpenAI Blog. Accessed: 2025-4-16.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Sheng-Hsuan Peng, Eric Michael Smith, Ivan Evtimov, Song Jiang, Pin-Yu Chen, Hongyuan Zhan, Haozhu Wang, Duen Horng Chau, Ma hesh Pasupuleti, and Jianfeng Chi. 2025. Large reasoning models learn better alignment from flawed thinking. *ArXiv*, abs/2510.00938.

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

Zhihong Shao, Yuxiang Luo, Chengda Lu, ZZ Ren, Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, and Xiaokang Zhang. 2025. Deepseekmath-v2: Towards self-verifiable mathematical reasoning. *arXiv preprint arXiv:2511.22570*.

Wentao Shi, Mengqi Yuan, Junkang Wu, Qifan Wang, and Fuli Feng. 2024. Direct multi-turn preference optimization for language agents. *Preprint*, arXiv:2406.14868.

Yuchong Sun, Che Liu, Kun Zhou, Jinwen Huang, Ruihua Song, Wayne Xin Zhao, Fuzheng Zhang, Di Zhang, and Kun Gai. 2024. Parrot: Enhancing multi-turn instruction following for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9729–9750.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

Yang Wan, Zheng Cao, Zhenhao Zhang, Zhengwen Zeng, Shuheng Shen, Changhua Meng, and Linchao Zhu. Mitigating conversational inertia in multi-turn agents through context bias calibration.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. 2025. Information gain-based policy optimization: A simple and effective approach for multi-turn llm agents. *arXiv preprint arXiv:2510.14967*.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. 2023. Mint: Evaluating llms in multi-turn interaction with tools and language feedback. *arXiv preprint arXiv:2309.10691*.

Zhepei Wei, Wenlin Yao, Yao Liu, Weizhi Zhang, Qin Lu, Liang Qiu, Changlong Yu, Puyang Xu, Chao Zhang, Bing Yin, and 1 others. 2025. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. *arXiv preprint arXiv:2505.16421*.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. 2025. Collabllm: From passive responders to active collaborators. *Forty-second International Conference on Machine Learning*.

Wei Xiong, Chengshuai Shi, Jiaming Shen, Aviv Rosenberg, Zhen Qin, Daniele Calandriello, Misha Khalman, Rishabh Joshi, Bilal Piot, Mohammad Saleh,

and 1 others. 2024. Building math agents with multi-turn iterative preference learning. *arXiv preprint arXiv:2409.02392*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, Yang Katie Zhao, and Mingyi Hong. 2025a. Reinforcing multi-turn reasoning in llm agents via turn-level credit assignment. In *ICML 2025 Workshop on Computer Use Agents*.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun MA, and Junxian He. 2025b. SimpleRL-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. In *Second Conference on Language Modeling*.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. 2025. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *Preprint*, arXiv:2503.15478.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. 2024. Archer: Training language model agents via hierarchical multi-turn rl. *arXiv preprint arXiv:2402.19446*.

A Additional Experiments

In this section, we present additional experiments to comprehensively investigate the nature of Contextual Inertia and validate the robustness of RLSTA. We begin by **quantifying the intrinsic vulnerability** of LLMs in multi-turn settings. We then extend our analysis of Contextual Inertia to include **more advanced models**. To further validate our approach, we examine the efficacy of RLSTA **without data filtering** and its generalization to reasoning-oriented **"Thinking" models**. Furthermore, we investigate the impact of our method on **single-turn capabilities** and corroborate our observations using **alternative evaluators** (gemini-2.5-pro (Comanici et al., 2025)). Finally, we demonstrate the robustness of our filtering strategy through its **transferability across heterogeneous data sources**.

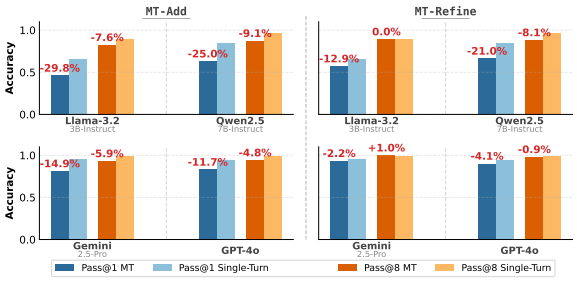

Figure 8: Vulnerability of LLMs in multi-turn settings: While Pass@1 drops significantly compared to single-turn, Pass@8 remains relatively stable. This indicates that the model retains the latent capability to solve the problem but is biased by the decoding path.

A.1 Vulnerability of LLMs in multi-turn settings

We begin by quantifying the vulnerability of LLMs in multi-turn settings. As illustrated in Figure 8, model performance exhibits substantial degradation in multi-turn settings compared to single-turn baselines. Specifically, the pass@1 performance suffers a precipitous drop, with an average decline of over 20% in the `MT-Add` and 10% in the `MT-Refine`. However, this degradation is largely mitigated when sampling is permitted: the pass@8 performance exhibits a much milder drop of less than 7% in `MT-Add` and merely 2% in `MT-Refine`. This discrepancy reveals a critical insight: the model retains the latent capability to solve the problem (evidenced by the preserved pass@8 score), but its most probable decoding trajectory is effectively "hijacked" by the conversation history.

A.2 Contextual Inertia Analysis

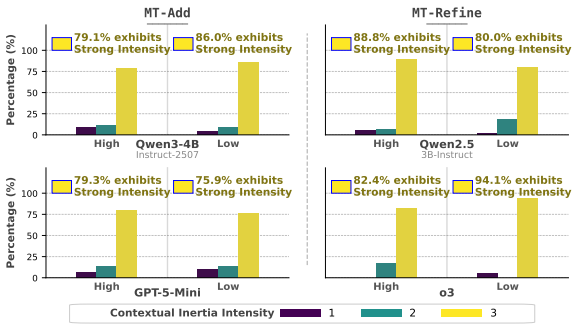

Figure 9: Additional experiments: the distribution of the contextual inertia intensity for more models.

We extend our behavioral analysis and root cause tracing to include advanced models, specifically GPT-5-mini (OpenAI, 2025a) and o3 (OpenAI, 2025b). As illustrated in Figure 9 and Figure 10, Indiscriminate nature of contextual inertia

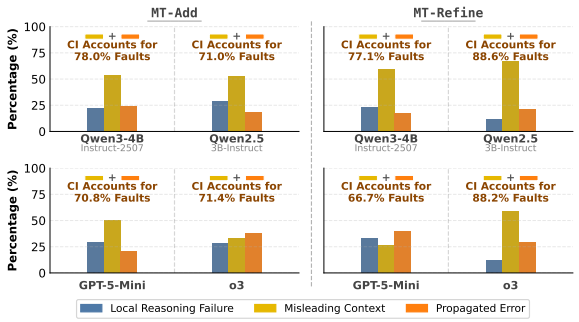

Figure 10: Additional experiments: Tracing the root cause of LLM failures.

persists across different models, even in reasoning-intensive models such as o3. Notably, over 70% of failures can be traced directly to errors in previous responses. These findings further corroborate our conclusions in Section 4, confirming that Contextual Inertia is the primary driver of LLM vulnerability in multi-turn interactions.

A.3 Effectiveness without Data Filtering

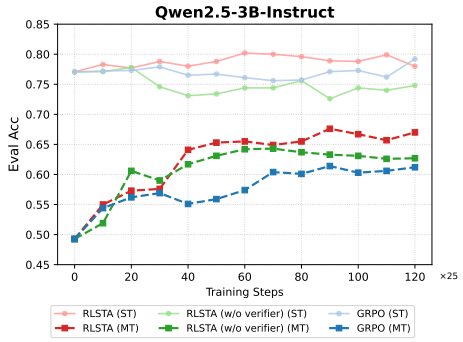

Figure 11: Training dynamics of Single-turn (ST) and Multi-turn (MT) performance on datasets without the filtering procedure.

RLSTA leverages the model's own strong reasoning capabilities to provide reward signals for multi-turn responses. While Section 6 demonstrated the effectiveness of RLSTA without external verifiers, those experiments utilized filtered data. Here, we conduct additional experiments on datasets without this filtering procedure, as shown in Figure 11. Although training on unfiltered data results in lower absolute performance compared to the filtered setting, RLSTA (w/o verifier) still achieves performance comparable to the GRPO baseline on the same data. This further demonstrates the effectiveness and generalizability of RLSTA, particularly in domains where data filtering or external verifiers are not feasible.

A.4 Generalization to Thinking Models

Table 5: Experimental results comparing Base and RLSTA on Qwen3-4B. The model was trained in *No Think* mode but evaluated in both *Think* and *No Think* modes. RLSTA yields consistent improvements across both inference settings.

Method	Qwen3-4B (No Think)				Qwen3-4B (Think)			
	MT-Add		MT-Refine	Avg	MT-Add		MT-Refine	Avg
	Math	Code	Math		Math	Code	Math	
Base	0.670	0.285	0.812	0.589	0.788	0.628	0.868	0.761
RLSTA	**0.801**	**0.333**	**0.873**	**0.669**	**0.920**	**0.790**	**0.941**	**0.884**

Finally, we investigate the applicability of RLSTA to modern reasoning models that utilize "Thinking" processes. We select Qwen3-4B (Yang et al., 2025) as the base model and train it using RLSTA in *No Think* mode. We then evaluate the model in both *Think* (reasoning enabled) and *No Think* modes during inference. The results, presented in Table 5, show that RLSTA significantly enhances performance in both modes. This suggests that RLSTA can effectively transfer the ability of breaking contextual inertia to reasoning-heavy paradigms, even when the training process does not explicitly target the thinking tokens.

A.5 Single-turn Performance

Table 6: Comparison of Base and RLSTA Performance Across Different Models. The Δ column indicates the relative difference in single-turn performance between Base and RLSTA.

Model	Base	RLSTA	Δ(%)
Llama-3.2-3B Instruct	0.663	0.684	+3.3
Qwen2.5-3b Instruct	0.771	0.763	-0.9
Qwen2.5-7b Instruct	0.853	0.856	+0.3
Qwen3-4B Instruct-2507	0.894	0.898	+0.4
Qwen3-4B (no think)	0.877	0.893	+1.8
Qwen3-4B (think)	0.898	0.890	-0.9
Average	**0.826**	**0.831**	**+0.7**

We evaluate the impact of RLSTA on single-turn capabilities on math, as shown in Table 6. The results demonstrate that RLSTA preserves the intrinsic single-turn capabilities of the base models while enhancing their multi-turn performance.

A.6 Contextual Inertia Intensity Using Gemini-2.5-pro

While we mainly analysis the contextual inertia intensity through gpt-4o (Figures 3,9,6), here we

provide additional analysis using Gemini-2.5-pro (Comanici et al., 2025), as shown in Figure 12, in most cases, the model's final answer m_n still exhibits strong inertia intensity to the preceding response m_{n-1}, and the distribution of this intensity remains indistinguishable regardless of the conversation history quality (high vs. low), which align with our observations using gpt-4o, further validate

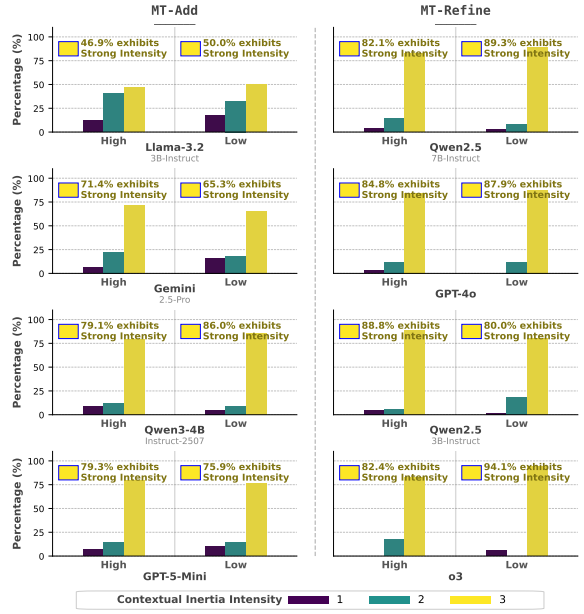

Figure 12: Distributions of Contextual Inertia Intensity $\mathtt{I}_{\mathrm{CI}}(m_n, m_{n-1})$. The inertia intensity are categorized using Gemini-2.5-pro.

A.7 Robustness of Latent Capability Filtering across Data Sources

We demonstrate that our Latent Capability Filtering strategy is not restricted to conversation histories generated by the target model $\mathcal{M}$ itself; it is equally effective when applied to raw histories from external models $\mathcal{M}'$. This transferability stems from the design of our filtering criterion (Eq. 2): it relies solely on the *target* model $\mathcal{M}$'s intrinsic single-turn capability. As long as $\mathcal{M}$ possesses the latent knowledge to solve the problem given the full context (i^{full}), it can provide a valid single-turn anchor reward (R_{s}) to guide the multi-turn generation, regardless of the origin of the conversation trajectory.

To validate this, we conduct cross-model experiments where Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct are trained using raw conversation data generated either by themselves (Self) or by the other model (Cross). The results are presented in Table 7.

As illustrated in Table 7, RLSTA exhibits stability across different data sources. Specifically,

Table 7: Cross-model evaluation of Latent Capability Filtering on different data sources. We compare the performance when using raw data generated by Qwen2.5-3B-Instruct versus Llama-3.2-3B-Instruct to train different target models.

Source of $\mathcal{D}_{\text{raw}}$	Qwen2.5-3B-Instruct				Llama-3.2-3B-Instruct			
	MT-Add		MT-Refine	Avg	MT-Add		MT-Refine	Avg
	Math	Code	Math		Math	Code	Math	
Qwen2.5-3B	0.715	0.256	0.745	0.572	0.649	0.205	0.640	0.498
Llama-3.2-3B	0.690	0.239	0.729	0.553	0.609	0.245	0.646	0.500

Qwen2.5-3B-Instruct maintains comparable average performance when trained on Llama-generated histories (0.553) versus its own histories (0.572). Similarly, Llama-3.2-3B shows negligible performance fluctuation (0.498 vs. 0.500). These findings suggest that our approach is robust to the data source, allowing models to effectively break contextual inertia even when learning from conversation trajectories generated by heterogeneous systems.

W2: Simplifying dialogues into just two isolated modes, MT-Add and MT-Refine, may be somewhat biased, as real-world interactions are often an interleaved combination of both.

A.8 Evaluation on Hybrid Multi-Turn Scenarios

While we primarily focus on `MT-Add` and `MT-Refine` as two fundamental multi-turn dialogue paradigms, real-world interactions often involve a complex interplay of both. To further evaluate the robustness of our method in more realistic, interleaved scenarios, we introduce two new hybrid settings based on the math datasets:

- **Add-then-Refine (AtR):** We simulate a scenario where the user first provides a sequence of corrupted information shards (similar to `MT-Add` but with noisy data), followed by a sequence of their corresponding corrections. This results in the interaction sequence $[s_1, \ldots, s_n, c_1, \ldots, c_n]$, representing a user who provides a complete set of initial, flawed details before issuing block corrections.
- **Add-Interleaved-with-Refine (AiwR):** We simulate a scenario where the user provides a corrupted shard s_i and immediately follows it with its correction c_i before moving to the next piece of information. This forms the interleaved sequence $[s_1, c_1, s_2, c_2, \ldots, s_n, c_n]$, where each element serves as a sequential query. This tests the model's ability to process immediate feedback

during continuous information intake.

We evaluated our RLSTA-tuned models trained exclusively on standard `MT-Add` and `MT-Refine` tasks—against the base models. The results, presented in Table 8, demonstrate that RLSTA generalizes well to these unseen hybrid structures. Despite not being explicitly trained on interleaved data, RLSTA yields substantial performance improvements over the base models in both AtR and AiwR settings.

Table 8: Evaluation of Base and RLSTA models on hybrid multi-turn scenarios (Add-then-Refine and Add-Interleaved-with-Refine). Models trained with RLSTA show consistent improvements on unseen hybrid interaction patterns.

Model	Setting	Base (Acc)	RLSTA (Acc)
Qwen2.5-3B-Instruct	AtR	0.476	**0.665**
	AiwR	0.471	**0.572**
Qwen2.5-7B-Instruct	AtR	0.568	**0.794**
	AiwR	0.542	**0.600**
Qwen3-4B-2507	AtR	0.574	**0.858**
	AiwR	0.615	**0.863**

B Additional Experimental Details

Training Details We implement our algorithms using the TRL library (von Werra et al., 2020). For Supervised Fine-Tuning (SFT), we set the learning rate to 1e-6 with a batch size of 32, selecting the checkpoint with the lowest validation loss over 5 epochs. For Direct Preference Optimization (DPO), we employ a learning rate of 5e-7 with a batch size of 16, training for 1 epoch. For both GRPO and RLSTA, we adopt a learning rate of 3e-7, a batch size of 16, KL penalty coefficient 1e-4, group size of 8, and temperature 1.0. Regarding training data, we sample 800 questions from the GSM8K dataset (Cobbe et al., 2021) and rewrite them into multiple shards. For data generation, following Peng et al. (2025), in Section 6, we utilize Qwen2.5-3B-Instruct to first generate raw multi-turn conversation histories during training, which are subsequently filtered specifically for each model. We consider both the `MT-Add` and `MT-Refine` scenarios. Specifically, for the Qwen series models (including Qwen2.5-3B/7B-Instruct and Qwen3-4B-Instruct-2507 / Qwen3-4B), we filter and retain 100 `MT-Refine` and 400 `MT-Add` conversation histories. For Llama-3.2-3B-Instruct, we retain 200 `MT-Refine` and 400 `MT-Add` conversation histories.

Evaluation Details Our evaluation follows the protocol of Laban et al. (2025). We run 8 simulations per instruction with a temperature of 0.7, reporting the average performance. The maximum token limit is set to 1,024 for No Think models and 8,192 for Qwen3-4B in thinking mode. We extract the final answer from the response using GPT-4o-mini for Math and Action tasks, while employing rule-based extraction for Code and Database tasks. During training, to reduce costs, we employ a simplified extraction strategy: we identify the last four numbers in the response and check whether the correct answer is present among them. Our results demonstrate that this simplified method effectively teaches the model to provide the correct final answer.

C Case Studies

In this section, we present a comparative case study between the base model and the RLSTA-aligned model. Both models are presented with an identical multi-turn conversation history containing a misleading intermediate reasoning trace. As illustrated below, the base model exhibits Contextual Inertia: it is misled by its own previous response, rigidly adhering to the hallucinated condition that 'insurance covers 80% of the cost,' which ultimately propagates into an incorrect final answer. In contrast, the RLSTA-tuned model successfully breaks this inertia. It filters out the invalid context from the previous turn, recalculates based solely on the user's verified constraints, and derives the correct total cost.

Case Study for Base model

Previous Information (Users):

- Calculate the total cost of John's physical therapy?
- John hurt his back and needed physical therapy.
- He attended therapy for 6 weeks, twice per week.
- Each session lasted 2 hours.

Last Assistant Response:

Assistant: Given that each session lasted 2 hours, we need to adjust our calculations to account for the duration of each session.
[...]
if the insurance covers 80% of the total cost, then John only pays 20% of the total cost.
[...]

Final Interaction:

User: the cost per hour for a session was $125.
Assistant: Given that the cost per hour for a session is $125, we need to adjust our calculations to account for the duration of each session.
[...]

Insurance covers 80%, so John pays 20%. Total cost after insurance : 0.2 times 3000 = 600

Therefore, the total cost of John's physical therapy after insurance coverage is 600

Case Study for Model after RLSTA

Previous Information (Users):

- Calculate the total cost of John's physical therapy?
- John hurt his back and needed physical therapy.
- He attended therapy for 6 weeks, twice per week.
- Each session lasted 2 hours.

Last Assistant Response:

Assistant: Given that each session lasted 2 hours, we need to adjust our calculations to account for the duration of each session.
[...]
if the insurance covers 80% of the total cost, then John only pays 20% of the total cost.
[...]

Final Interaction:

User: the cost per hour for a session was $125.
Assistant: Let's solve the problem step by step.
[...]
Perform the multiplication: 24 times 125 = 24 times (100 + 25) = (24 times 100) + (24 times 25) 24 times 100 = 2400 24 times 25 = 600 2400 + 600 = 3000

Therefore, the total cost of John's physical therapy is

3000

D Prompts

D.1 Similarity Analysis

We employ GPT-4o to assess the logical similarity between response pairs, focusing specifically on their reasoning structures and key derivation steps. We quantify this similarity on a 5-point scale. For the visualizations presented in Figure 3, Figure 6, and Figure 9, we categorize these scores into three distinct tiers represented by a color gradient: **High Similarity** (Levels 4–5), **Medium Similarity** (Level 3), and **Low Similarity** (Levels 1–2).

```
EVALUATION_SYSTEM_PROMPT = """You are a Senior Logic
      Analyst specializing in auditing multi-turn AI
      reasoning and Chain-of-Thought (CoT)
     consistency.

Your objective is to evaluate how well an AI
     assistant adapts its logic to changing
     constraints while maintaining mathematical
     precision. You must perform a deep structural
     comparison between the provided Example, First
     Turn, and Second Turn.

Key Responsibilities:
**Verify Accuracy:** Rigorously check the final
     answer against the Ground Truth.

Output your analysis in strict JSON format. Do not
     include markdown formatting (like ```json) or
     conversational text."""

EVALUATION_USER_PROMPT_TEMPLATE = """
## Inputs

### Example Answer Attempt (example_answer_attempt):
{example_answer_attempt}

### First Turn
**Query (first_turn_query):**
{first_turn_query}

**Assistant Response (first_turn_response):**
{first_turn_response}

### Second Turn
**Query (second_turn_query):**
{second_turn_query}

**Assistant Response (second_turn_response):**
{second_turn_response}

### Ground Truth
**Correct Answer (correct_answer):**
{correct_answer}

---

## Evaluation Instructions

### Multi-Turn Response Similarity Assessment
Analyze the similarity between responses, focusing
     on logic structure and Chain of Thought (CoT)
     steps. Provide a similarity score (1-5) for the
      following pairs:
1. **example-r1**: Similarity between `
     example_answer_attempt` and `
     first_turn_response`.
2. **example-r2**: Similarity between `
     example_answer_attempt` and `
     second_turn_response`.
3. **r1-r2**: Similarity between `
     first_turn_response` and `second_turn_response
     `.

**Similarity Score Definitions:**
- **1 (Distinct):** The response uses a completely
     fresh approach; no apparent similarity in logic
      or structure.
- **2 (Minimal):** The response is mostly
     independent; shares only superficial elements
     but uses different reasoning.
- **3 (Moderate):** The response reuses some steps
     or general methods but includes meaningful
     adaptations to the CoT structure.
- **4 (Strong):** The response heavily reuses the
     CoT structure and steps, with limited
     adaptation (e.g., simply changing numbers while
      keeping the exact same logic flow).
- **5 (Near-identical):** The response mirrors the
     reference with minimal changes, using almost
     the same logic structure, steps, and phrasing.
---

## Output Format

Return exactly one JSON object matching this schema:

```json
{{
  "similarity_assessment": {{
  "analysis process - example-r1": "<Detailed
       explanation of the similarity assessment>",
  "example-r1": 1 | 2 | 3 | 4 | 5,
  "analysis process - example-r2": "<Detailed
       explanation of the similarity assessment>",
  "example-r2": 1 | 2 | 3 | 4 | 5,
  "analysis process - r1-r2": "<Detailed explanation
       of the similarity assessment>",
  "r1-r2": 1 | 2 | 3 | 4 | 5
  }},
  "notes": "string"
}}

Return only the JSON object with no additional text
     or markdown formatting.
"""
```
```

D.2 Data Preparation

We directly adapt the prompts in Laban et al. (2025) to process a single-turn instruction in GSM8K into shards. The specific prompts used for segmentation and rephrasing are provided below:

```
Segmentation_Prompt_Format = """
You are a given a arithmetic question, and your task is to segment the question into units of information that each reveal a single piece of information of the question.
You must output a list of segments in the following JSON format:
[
    {"segment": "[exact excerpt from the question]", "is_required": 1|0},
    {"segment": "[exact excerpt from the question]", "is_required": 1|0},
    ...
]

Rules:
- [is_required] For each segment, you must specify whether this particular segment is required (1) or not strictly necessary (0) to answer the question.
- [Non-overlapping] The segments must be non-overlapping and cover the entire question. You can optionally leave some gaps for non-essential portions of the original question (delimiters, headers, etc.)
- [Minimalistic] You should split the information in the segments to as small as possible. If you have a compound expression (X and Y), you should split it into two segments. Each segment should represent a unit of information.
- [Valid Segments] Only extract segments from the text of the question. Do not include example inputs/outputs as segments.
- [Segment count] The number of segments should not be more than 10.

Example Question:

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?


Output:
{"segments": [
    {"segment": "There are 15 trees in the grove", "is_required": 1},
    {"segment": "Grove workers will plant trees in the grove today", "is_required": 1},
    {"segment": "After they are done, there will be 21 trees", "is_required": 1},
    {"segment": "How many trees did the grove workers plant today?", "is_required": 1},
]}

Now complete the task for the following fully specified question:

"""


Rephrasing_Prompt_Format = """
You are given a segment of a complete question, and your task is to: (1) choose one that will be the initial query of a multi-step query, and then each of the remaining segment should be one int provided to the system in a follow-up turn of the conversation.

Your output should be a JSON object in the following format:
{
    "initial_segment": "[exact excerpt from the question]",
    "initial_query": "conversational version of the initial segment",
    "hints": [
    {"segment": "[exact excerpt from the question]", "hint": "conversational version of the segment taking the rest of the question into account"}
    ]
}

Example:

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?

Segments:
[
    {"segment": "There are 15 trees in the grove", "is_required": 1},
    {"segment": "Grove workers will plant trees in the grove today", "is_required": 0},
    {"segment": "After they are done, there will be 21 trees", "is_required": 1},
    {"segment": "How many trees did the grove workers plant today?", "is_required": 1},
]

Output:
{
    "initial_segment": "How many trees did the grove workers plant today?",
    "initial_query": "I need to calculate the number of trees planted by the grove workers today",
    "hints": [
        {"segment": "There are 15 trees in the grove", "hint": "15 trees are in the grove before planting"},
        {"segment": "Grove workers will plant trees in the grove today", "hint": "Grove workers will plant trees in the grove today"},
        {"segment": "After they are done, there will be 21 trees", "hint": "I see 21 trees after the grove workers are done"},
    ]
}

Rules:
- [Query selection] Choose the question segment from the segments to form the initial query.
- [Transform each segment] Make sure each segment is included either as the initial query or as a hint. Do not forget any segments.
- [Short initial query] Make the initial query short, not a full sentence, similar to how users use a search engine like Google
- [Order of hints] Order the hints in order of importance, from most to least important to the query. You do not need to keep the order the segments are provided in.

Now complete the task for the following fully specified question and segments:

"""
```

D.3 Abstain Prompt

While our standard evaluation follows the default system prompt protocol outlined in Laban et al. (2025), we introduce a variant setting to facilitate a direct comparison with RLAAR, an abstention-based strategy. Specifically, we evaluate our method with an additional system instruction that explicitly encourages the model to withhold answers when information is incomplete (denoted as RLSTA + Abstain). The specific prompt used for this setting is provided below:

```
[Important] I will provide additional conditions
     incrementally in turns rather than all at once.
      Do not provide a concrete answer attempt you
     have all necessary details.
After each new condition, review your previous
     response to ensure it remains correct and
     compatible.
Update your solution accordingly while maintaining
     correctness under all conditions provided so
     far.
```

D.4 Corrupting Prompt

In the `MT-Refine` task, we employ GPT-4o to systematically alter specific details in the information shards to corrupt the initial query. The prompt used for this process is as follows:

```
prompt = f"""
You are a data augmentation expert.
Below is a list of "shards" representing parts of a
     math problem.

Your Task:
1. I have excluded the first shard (the main
     question).
2. You must modify **EVERY** single shard provided
     in the list below.
3. Apply the following modification style:
    **STRICT NUMERIC MODIFICATION**:
    - Identify the numbers/integers in the text.
    - Change the values of these numbers to
        different reasonable values.
    - Do NOT change the words, units (years, fruits)
        , or the logic (multiplication, addition).
    - ONLY change the digits.

Input Data (Shards to modify):
{shards_text}

Output Format:
Return a JSON object with a single key "
     modified_shards" containing the list of
     modified objects.
Example:
{{
  "modified_shards": [
      {{ "shard_id": 2, "shard": "modified text..."
          }},
      {{ "shard_id": 3, "shard": "modified text..."
          }}
  ]
}}
"""
```