When Thoughts Meet Facts: Reusable Reasoning for Long-Context LMs

Anonymous ACL submission

Abstract

Recent Long-Context Language Models (LCLMs) can process hundreds of thousands of tokens in a single prompt, enabling new opportunities for knowledge-intensive multi-hop reasoning by integrating large sets of retrieved documents or, in some cases, directly all necessary information. However, simply feeding more documents into the context window fails to capture how evidence should be connected. We address this gap with *thought templates*, reusable reasoning patterns derived from prior problem solving that structure how evidence is combined and guide multi-hop inference alongside factual documents. To keep these templates effective, we propose an update strategy that iteratively refines templates derived from training data through natural-language feedback. Across diverse benchmarks and LCLM families, our approach delivers consistent gains over strong baselines in both retrieval-based and retrieval-free settings. Furthermore, we show that optimized templates can be distilled into relatively smaller open-source models, demonstrating its broad applicability. We refer to our framework as Thought Template Augmented LCLMs (ToTAL).

1 Introduction

Knowledge-intensive multi-hop reasoning tasks require models to gather evidence from multiple documents, and combine it through reasoning (Trivedi et al., 2022, 2023; Tang and Yang, 2024; Huang et al., 2025). These tasks are difficult because relevant evidence must not only be identified, but also be connected in a structured way, requiring knowledge-based reasoning. The standard solution, Retrieval-Augmented Generation (RAG), tackles this by first retrieving a small set of relevant documents and then generating an answer from them (Lewis et al., 2020; Jeong et al., 2024).

The rise of Long-Context Language Models (LCLMs) has shifted this paradigm by enabling prompts of hundreds of thousands of tokens (Anil et al., 2023; Comanici et al., 2025; Anthropic, 2025; OpenAI, 2025a). This advancement makes it possible to "just put everything into the prompt," such as feeding in all retrieved documents (Lee et al., 2025) or many in-context examples (Agarwal et al., 2024; Baek et al., 2025). Compared to conventional RAG, which risks cascading errors from retrieval, LCLMs support a one-step formulation that mitigates such errors, and in some domains (e.g., enterprise settings) can even absorb an entire document collection into the prompt. However, increasing recall with more documents alone remains insufficient, since models may struggle to connect pieces of evidence. Existing work on LCLMs has largely focused on scaling input size rather than strengthening reasoning, leaving this gap unaddressed.

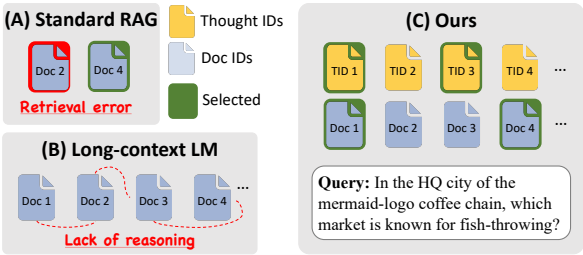

Figure 1: Thoughts and facts in LCLM, compared to transitional RAG and simple stuffing in LCLM.

While one possible direction is adopting reasoning strategies such as Chain-of-Thought (Wei et al., 2022), which elicit step-by-step reasoning, it remains ad-hoc and query-specific, and they are not designed to cope with the vast, document-heavy contexts enabled by LCLMs. To address this, we introduce *thought templates*: reusable reasoning patterns (or epistemic knowledge from prior experience) that act as structured scaffolds for integrating and organizing evidence in these long-context settings. Templates act as a cache of prior reasoning behaviors capturing *how to think*, while documents provide the factual content capturing *what to know*. Importantly, although the entire template set is sup-

plied, LCLMs selectively leverage on the relevant ones for each query, thereby enabling compositional reasoning over complex evidence.

To instantiate this idea, we automatically construct templates from multi-hop QA datasets in a compositional manner, allowing LCLMs to flexibly recombine multiple templates within a single generation. Unlike prior approaches, which retrieve a single problem-specific reasoning trace (Yang et al., 2024b, 2025a), our method enables reusability across queries. This compositional design also improves performance by allowing LCLMs to generalize to more complex reasoning tasks. To further improve effectiveness, we treat thought templates as external parameters of LCLMs and refine them iteratively using natural language feedback. Feedback derived from model errors specifies how templates should be revised, functioning like a gradient update but without altering model weights.

We present a framework, Thought Template Augmented LCLMs (ToTAL), that equips long-context models with reusable reasoning patterns and iteratively refines them through natural language feedback. We validate ToTAL on diverse knowledge-intensive datasets that require both factual grounding and multi-hop reasoning. Furthermore, we evaluate it in two settings: an idealized setup without retrieval and a more practical scenario with retrieval. Across both settings, thought templates consistently boost LCLM performance, and our feedback-driven update strategy yields additional gains. These results highlight the promise of equipping LCLMs with structured reasoning patterns rather than relying solely on larger contexts.

2 Proposed Method

Our method is motivated by three observations: (1) simply increasing the number of accessible documents in LCLMs does not guarantee better reasoning; (2) current models often lack explicit, structured strategies for combining evidence across multiple steps; and (3) once distilled, such strategies can be generalized and reused across models. Below we introduce the necessary background and describe the design of ToTAL.

2.1 Preliminaries

We first outline the challenges and the limitations of existing paradigms of multi-hop reasoning.

Knowledge-intensive Multi-hop Reasoning Multi-hop reasoning requires gathering and integrating evidence scattered across multiple documents and composing intermediate steps for the final answer. Formally, given a query $\boldsymbol{q}$ and a large corpus of documents $\mathcal{D} = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_N\}$, the objective is to generate the correct answer $\boldsymbol{a}$ by selecting a relevant evidence subset $\mathcal{D}_q \subseteq \mathcal{D}$ and chaining reasoning steps over it.

Retrieval-Augmented Generation (RAG) Conventional approaches rely on RAG: a retriever first identifies a subset of documents $\mathcal{D}_q = \texttt{Retriever}(\boldsymbol{q}, \mathcal{D})$, and then a Language Model (LM) generates an answer conditioned on both $\boldsymbol{q}$ and $\mathcal{D}_q$, denoted as $\boldsymbol{a} = \texttt{LM}(\boldsymbol{q}, \mathcal{D}_q)$. Since earlier LMs were limited by context length, retrieval quality was crucial: poor retrieval caused cascading errors by omitting essential evidence.

Long-Context Language Models (LCLM) Recent LCLMs can process even millions of tokens in a single prompt, thereby allowing the direct inclusion of large evidence sets (or entire corpora) into the context: $\boldsymbol{a} = \texttt{LCLM}(\boldsymbol{q}, \mathcal{D})$. Alternatively, retrieval still can be used to select a much larger set of documents than before, $\mathcal{D}_q^* = \texttt{Retriever}(\boldsymbol{q}, \mathcal{D})$, where $|\mathcal{D}_q| \ll |\mathcal{D}_q^*|$. Thus, LCLM supports two regimes: inserting full corpus $\mathcal{D}$, or large retrieved subset $\mathcal{D}_{\texttt{large}}$ where $\mathcal{D}_{\texttt{large}} \in \{\mathcal{D}, \mathcal{D}_q^*\}$. However, simply scaling document access is insufficient: the bottleneck now lies in how to *structure* and *reuse* reasoning over abundant knowledge. At the same time, finetuning LCLM to explicitly learn long reasoning chains is often infeasible due to their high cost and limited accessibility.

2.2 Thought Template Augmented LCLMs

To bridge these gaps, we introduce ToTAL, a framework that enables better reasoning in large document contexts *without any model finetuning* by leveraging *thought templates* – structured thought processes built from training data and refined iteratively through textual gradient feedback (Figure 2). These updated templates then guide the LCLM in organizing evidence and performing multi-step reasoning more effectively during inference.

Thought Templates A *thought template* is a reusable high-level reasoning pattern, distilled from prior problem-solving. Each template provides a structured outline of intermediate steps that can be instantiated for new queries. Formally, let

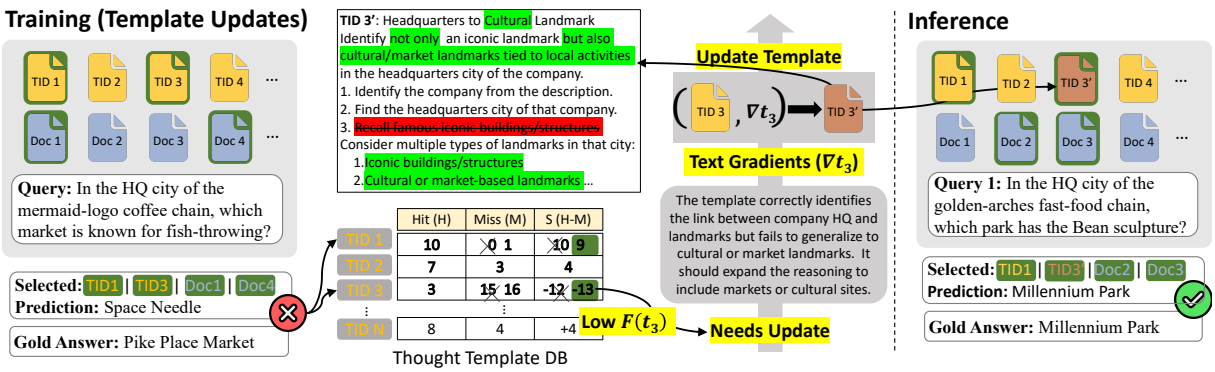

Figure 2: Illustration of training and inference stages for template updates. Low-performing templates are identified via hit/miss statistics and refined with textual gradient feedback, enabling improved performance on new queries during inference.

$\mathcal{T} = \{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_m\}$ denote the set of templates. At inference, the LCLM is conditioned on both the query $\boldsymbol{q}$, large evidence set $\mathcal{D}_{\texttt{large}}$, and templates:

$$\hat{\boldsymbol{a}} = \texttt{LCLM}(\boldsymbol{q}, \mathcal{T}, \mathcal{D}_{\texttt{large}})$$

Template Construction To build the initial template set $\mathcal{T}$, we prompt an LCLM to generate templates, conditioned on the training queries $\boldsymbol{q}_{\texttt{train}}$, their gold answers $\boldsymbol{a}_{\texttt{train}}$, and optionally solution paths $\boldsymbol{s}_{\texttt{train}}$ from the training set:

$$\boldsymbol{t}_i = \texttt{LCLM}(\boldsymbol{q}_{\texttt{train}}, \boldsymbol{a}_{\texttt{train}}, [\boldsymbol{s}_{\texttt{train}}])$$

This procedure is inspired by Yang et al. (2025a), who also derive templates using LLMs. However, instead of capturing full example-specific solutions, we decompose them into sub-templates that are reusable across queries and generalize more effectively. At inference, the model selectively applies and composes relevant templates from $\mathcal{T}$ with the query and supporting documents. Below shows an example of a thought template $\boldsymbol{t}_3$ generated from the following template construction process.

> **TID 3:** Headquarters to Landmark
> Identify an iconic landmark in the headquarters city of the company.
> 1. Identify the company from the description.
> 2. Find the headquarters city of that company.
> 3. Recall famous iconic buildings/structures ...

Template Update Strategy Initial templates may be noisy or suboptimal. Thus, we iteratively refine templates $\boldsymbol{t}_i$ using natural-language feedback as a *surrogate gradient*. We first assign each template $\boldsymbol{t}_i$ an explicit performance score $F(\boldsymbol{t}_i)$ to compute which reasoning patterns contribute positively or negatively to model outputs. Specifically, for each $\boldsymbol{q}_{\texttt{train}}$ with $\boldsymbol{a}_{\texttt{train}}$, we obtain the model's prediction $\hat{\boldsymbol{a}}_{\texttt{train}} = \texttt{LCLM}(\boldsymbol{q}_{\texttt{train}}, \mathcal{T}, \mathcal{D}_{\texttt{large}})$. Then, $\boldsymbol{t}_i$ is assigned an aggregated score as:

$$F(\boldsymbol{t}_i) = \sum_{\boldsymbol{q}_{\texttt{train}}} f_i(\boldsymbol{q}_{\texttt{train}}),$$

where $f_i(\boldsymbol{q}_{\texttt{train}})$ measures the performance of $\boldsymbol{t}_i$ on $\boldsymbol{q}_{\texttt{train}}$ by comparing $\hat{\boldsymbol{a}}_{\texttt{train}}$ with $\boldsymbol{a}_{\texttt{train}}$ (e.g., using metrics such as exact match or F1 in QA tasks). Importantly, $f_i(\boldsymbol{q}_{\texttt{train}})$ is computed only for queries where $\boldsymbol{t}_i$ is actually selected. Templates with scores below a threshold $F(\boldsymbol{t}_i) < \tau$[1] are identified as low-performing and selected for refinement (e.g., TID 3 in the template database in Figure 2). This enables targeted refinement, updating only low-performing templates, while maintaining the stability of well-performing ones.

For each low-performing template, another LM analyzes its failure cases by comparing the query $\boldsymbol{q}_{\texttt{train}}$, its prediction $\hat{\boldsymbol{a}}_{\texttt{train}}$, the gold answer $\boldsymbol{a}_{\texttt{train}}$, and the applied template $\boldsymbol{t}_i$, and produces a natural-language "textual gradient" feedback:

$$\nabla \boldsymbol{t}_i = \texttt{LM}_{\texttt{Feedback}}(\boldsymbol{q}_{\texttt{train}}, \hat{\boldsymbol{a}}_{\texttt{train}}, \boldsymbol{a}_{\texttt{train}}, \boldsymbol{t}_i)$$

Below is an example feedback $\nabla \boldsymbol{t}_3$.

> ∇ **TID 3:** The template correctly identifies the link between company HQ and landmarks but fails to generalize cultural or market landmarks. It should expand the reasoning to include..

This textual gradient is accompanied by a discrete decision indicating the appropriate update action:

$$d_i \in \{\textsc{Keep}, \textsc{Fix}, \textsc{Add}, \textsc{Discard}\}$$

For KEEP, the template remains unchanged, while for DISCARD, the template is removed. For FIX and ADD, $\nabla \boldsymbol{t}_i$ is passed to another LM:

$$\boldsymbol{t}_i' = \texttt{LM}_{\texttt{update}}(\boldsymbol{t}_i, \nabla \boldsymbol{t}_i)$$

[1] τ denoting a threshold selected with the validation set

This iterative refinement process progressively improves the template set $\mathcal{T}$, which is subsequently used during inference to guide reasoning. The updated $\boldsymbol{t}_3'$ updated from $\boldsymbol{t}_3$ looks like below:

> **TID 3':** Headquarters to Cultural Landmark
> Identify not only an iconic landmark but also cultural/market landmarks tied to local activities in the headquarters city of the company.
> 1. Identify the company from the description.
> 2. Find the headquarters city of that company.
> 3. ~~Recall famous iconic buildings/structures~~ ...

3 Experimental Setup

3.1 Datasets

We evaluate TOTAL on four challenging multi-hop QA benchmarks: MuSiQue (Trivedi et al., 2022), CRAG (Yang et al., 2024c), FanOutQA (Zhu et al., 2024), and Housing QA (Zheng et al., 2025). MuSiQue requires reasoning over multiple passages and is widely used for evaluating multi-hop question answering. CRAG focuses on diverse and dynamic queries, going beyond traditional datasets by incorporating less popular topics and more complex reasoning types. FanOutQA consists of long-context Wikipedia documents. Housing QA evaluates domain-specific legal queries that require retrieving and reasoning over statutory texts.

3.2 Baseline Models

We compare TOTAL against four representative baselines, all of which use LCLMs as base models:

- **NAÏVE:** Directly generates answers from the query without any auxiliary context.
- **CHAIN-OF-THOUGHT (COT)** (Kojima et al., 2022): A prompting-based reasoning approach using the phrase "Let's think step by step."
- **CORPUS-IN-CONTEXT (CIC)** (Lee et al., 2025): Leverages the extended context window of LCLMs by inserting the entire documents directly into the prompt.
- **CIC + COT:** Combines CIC with COT, aiming to jointly utilize large-context access and explicit reasoning cues.

TOTAL differs from these baselines by introducing structured, reusable reasoning patterns (*thought templates*) that guide LCLMs to organize and apply evidence effectively, without additional fine-tuning.

3.3 Implementation Details

We evaluate across a diverse suite of LCLMs, including proprietary frontier models, Claude-Sonnet 4 (Anthropic, 2025), and Gemini 2.5 Flash (Comanici et al., 2025), and GPT-4.1 (OpenAI, 2025b) as well as open-source LLMs such as OSS (120B) (Agarwal et al., 2025) and DeepSeek-R1 (Guo et al., 2025). For retrieval-based settings, we employ BM25 (Robertson et al., 1994) as the retriever. We adopt standard QA metrics tailored to each dataset: F1 score for MuSiQue, CRAG, and FanOutQA, and Accuracy for Housing QA with binary outputs, and use the same metrics to compute template scores. Unless stated otherwise, we primarily use Claude on MuSiQue for analyses. We provide the prompts for template construction and update in Figures 15, 16, and 17.

3.4 Data Processing

For MuSiQue, we use the 128k-token version from the LOFT benchmark (Lee et al., 2025), and apply a similar preprocessing procedure to the other datasets, including matching the number of test queries. For CRAG, we focus on the *Multi-hop* and *Post-processing heavy* categories to target complex reasoning cases. For both CRAG and Housing QA, we construct corpora by aggregating all relevant snippets or statutes for each query, capped at 128k tokens. For FanOutQA, since its context units are full Wikipedia pages, we build a query-specific corpus containing only documents relevant to each question, also truncated to 128k tokens.

For the retrieval setting, we first construct 1M-token corpora for MuSiQue, CRAG, and Housing QA, and then subsample them to match the 128k-token budget (800 documents out of 6,650 for MuSiQue, 300 out of 2,307 for CRAG, and 480 out of 5,924 for Housing QA). For FanOutQA, we use all 2,142 documents from the original corpus as the retrieval corpus, and then retrieve 5 documents.

Regarding the template design, we construct the initial template set by sampling 50 QA pairs from the training data, ensuring no overlap with the test queries. For the template update strategy, we also use another subset of the training samples and determine the threshold τ on the validation set.

4 Results and Analyses

4.1 Main Results

Table 1 presents the performance across all benchmarks. Although recent LCLMs demonstrate

Table 1: Main results on four multi-hop reasoning datasets under the LCLM setting. We report F1 on MuSiQue, CRAG, and FanOutQA, Accuracy on Housing QA, and the overall Average. The best results are highlighted in **bold**.

	Methods	MuSiQue (F1)	CRAG (F1)	FanOutQA (F1)	Housing QA (Acc.)	Average
Claude	NAÏVE	27.57 ± 0.27	20.49 ± 1.02	46.72 ± 1.15	60.33 ± 2.08	38.78
	COT	28.10 ± 0.91	20.32 ± 1.29	45.54 ± 0.13	57.67 ± 2.08	37.90
	CIC	63.87 ± 0.91	17.32 ± 0.57	63.74 ± 1.50	71.67 ± 0.58	54.15
	CIC + COT	65.07 ± 0.16	18.86 ± 0.01	66.29 ± 0.53	75.00 ± 2.00	56.30
	TOTAL (Ours)	**73.30 ± 1.24**	**30.08 ± 0.83**	**69.99 ± 1.61**	**82.67 ± 0.58**	**64.01**
Gemini	NAÏVE	25.48 ± 1.85	22.03 ± 1.31	46.54 ± 1.85	58.00 ± 2.00	38.01
	COT	23.03 ± 0.80	24.62 ± 0.74	43.54 ± 1.54	58.67 ± 2.89	37.46
	CIC	66.54 ± 1.10	25.45 ± 0.68	66.44 ± 1.33	68.33 ± 1.15	56.69
	CIC + COT	67.17 ± 1.01	25.77 ± 0.30	66.97 ± 0.48	70.33 ± 0.58	57.56
	TOTAL (Ours)	**72.86 ± 0.71**	**27.71 ± 0.51**	**71.84 ± 0.62**	**74.33 ± 0.58**	**61.68**
GPT	NAÏVE	32.43 ± 0.67	25.73 ± 0.62	48.77 ± 1.41	60.33 ± 0.58	41.81
	COT	32.39 ± 0.15	23.24 ± 0.89	49.09 ± 0.97	61.33 ± 0.58	41.51
	CIC	63.79 ± 0.95	22.12 ± 0.69	63.39 ± 1.23	64.33 ± 0.58	52.50
	CIC + COT	65.11 ± 0.44	21.72 ± 0.75	66.35 ± 0.62	66.00 ± 1.00	54.79
	TOTAL (Ours)	**66.38 ± 0.13**	**26.31 ± 0.74**	**69.07 ± 1.83**	**70.00 ± 1.00**	**57.94**

Table 2: RAG results of LCLMs with retrieved documents.

Methods	MSQ.	CRAG	FOQA	HQA
CIC	41.63	13.10	26.57	70.00
TOTAL (Ours)	**47.90**	**19.87**	**32.16**	**76.50**

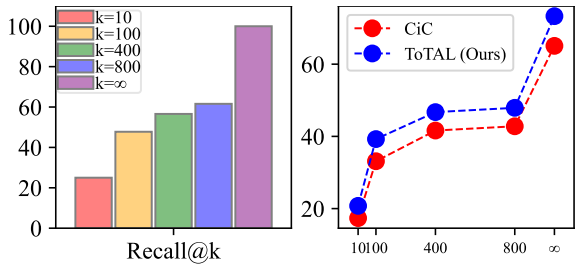

Figure 3: RAG results on MuSiQue, showing retrieval recall at different k values (left) and QA performance (F1) (right).

strong capabilities, they still struggle with complex and knowledge-intensive multi-hop queries. This is reflected in the performance of the NAÏVE baseline, which lacks access to external facts and relies on the internalized knowledge. Similarly, CHAIN-OF-THOUGHT (COT) prompting yields only marginal improvements, suggesting that explicitly eliciting step-by-step reasoning alone is insufficient for multi-hop knowledge integration. CORPUS-IN-CONTEXT (CIC) improves performance by leveraging the extended context window of LCLMs to include the entire corpus in the prompt. However, its gains remain limited since it treats the task as evidence aggregation rather than reasoning composition. Even when combined with COT (CIC + COT), improvements are modest. In contrast, TOTAL introduces implicit reasoning structure through reusable templates, *consistently outperforming all baselines across datasets*. This highlights the value of guiding LCLMs with structured reasoning patterns rather than relying solely on surface-level prompting strategies.

4.2 When Partial Context Given

While LCLMs are capable of handling large contexts, retrieval becomes essential when the full corpus cannot be included. We evaluate this retrieval-augmented scenario by comparing TOTAL with CIC, with both models given the same retrieved documents under our full-context budget. Table 2 shows that TOTAL consistently outperforms CIC, demonstrating that reasoning templates provide complementary advantages even in retrieval settings. To further examine this scenario, we vary the number of retrieved documents (k) to emulate more realistic retrieval-augmented scenarios in Figure 3. As the number of retrieved documents increases, both retrieval recall and QA performance improve, confirming that *long-context models benefit from expanded evidence access*. However, when compared with the idealized setting where all documents are available to the LCLM ($k = \infty$), retrieval still imposes a bottleneck. Importantly, our template-based approach consistently enhances performance across all retrieval sizes, demonstrating its robustness and adaptability. As LCLMs continue to scale in context length, such structured reasoning strategies are expected to further amplify their benefits.

4.3 Effectiveness of Template Update Strategy

To assess the contribution of the template update strategy, we report ablation results in Figure 4. The

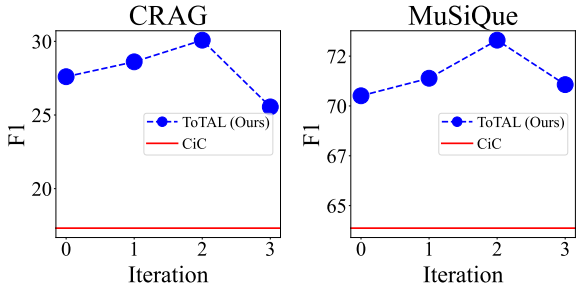

Figure 4: Iteration results of updates on CRAG and MuSiQue.

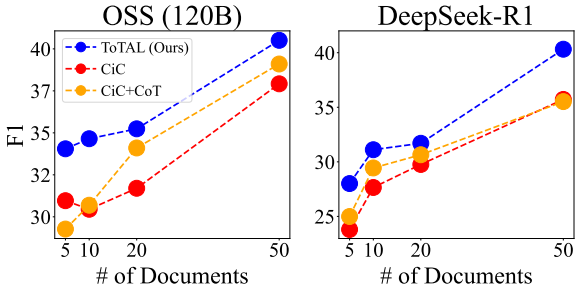

Figure 5: Generalization of templates to open-source models.

Table 3: Transferability of templates across LCLMs, where templates generated by GPT and Gemini are applied to Claude.

Methods	Source → Target	F1
CiC	–	63.87
ToTAL (Ours)	Gemini → Claude	**70.94**
	GPT → Claude	**70.11**

Table 4: Performance when using templates generated by open-source models versus distilled from a frontier model.

Methods	OSS	DeepSeek-R1
Naïve	17.27	14.98
CiC	30.45	27.65
ToTAL w/ Open	32.44	29.53
ToTAL w/ Distilled	**34.65**	**31.11**

Table 5: Results without compositionality and with oracle.

Methods	F1
CiC	63.87
ToTAL (Ours)	73.30
ToTAL w/o Compositional Templates	67.80
ToTAL w/ Oracle Templates	**78.49**

results show clear performance gains when applying iterative updates over the initial template set. *Refining low-performing templates via feedback substantially enhances reasoning accuracy*, validating our design choice of using natural-language feedback as a surrogate optimization signal. Even without updates, the initial template set already outperforms CiC, indicating that structured reasoning guidance itself contributes significant benefits. Performance plateauing around the second iteration reflects a diminishing returns effect, commonly observed in conventional ML, suggesting that the updates have effectively converged. We further show the updates across iterations in Appendix A.2.

4.4 Transferability of Templates

We evaluate generalization by testing template transfer across frontier models and applying templates from Claude to open-source LLMs. Table 3 shows that templates from one frontier LCLM can be effectively applied to others. Similarly, Figure 5 shows that these templates also transfer well to open-source models, yielding consistent improvements over baselines even under shorter context windows and retrieval-augmented settings. The gains remain stable across varying top-k values, underscoring that *thought templates encode model-agnostic reasoning structures* that generalize across architectures and retrieval conditions.

Templates Generated by Open-Source LLMs. We further investigate whether templates can be generated and refined entirely by open-source models. In Table 4, templates produced and updated by the same open-source model already surpass the CiC baseline, confirming the feasibility of a fully open pipeline. While templates derived from frontier LCLMs achieve higher performance, results suggest that template quality scales with model capacity, yet open-source systems can still produce competitive and practical reasoning patterns.

4.5 Impact of Template Quality

To get a more generalizable template, we design it compositionally, decomposing reasoning into multiple sub-templates rather than a single holistic one encompassing all steps. Table 5 shows that removing compositionality causes a measurable performance drop, confirming that smaller, modular templates promote better generalization across queries. In addition, to estimate the upper bound of our framework, we evaluate **oracle** templates constructed directly from *test queries*. The results show that oracle templates achieve substantially higher scores, representing the potential performance ceiling achievable with perfect template design. The performance gap between oracle and learned templates highlights interesting and promising directions for future work, such as automatic template search and meta-learning strategies for reasoning refinement.

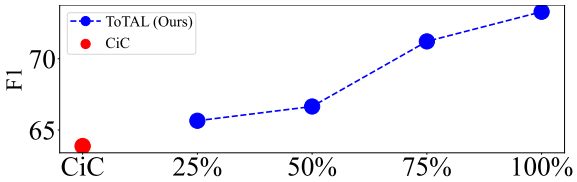

Figure 6: Varying the percentage of templates on MuSiQue.

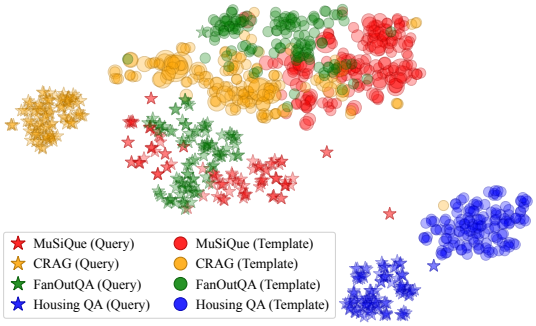

Figure 7: TSNE of the queries and templates, using embeddings from Sentence-BERT (Reimers and Gurevych, 2019).

4.6 Impact of Template Quantity

We also study how the number of templates affects performance by sampling different proportions (bottom 25%, 50%, and 75%) of the template pool based on their scores. As shown in Figure 6, performance remains competitive even with only 25% of the templates, and continues to improve as more templates are included. This suggests that high-scoring templates encode broadly reusable reasoning patterns, enabling strong performance even when the template set is reduced, although including the full set yields the best overall results.

4.7 Template Analyses Beyond Performance

Template–Query Clustering. As illustrated in Figure 7, queries and their associated templates form coherent clusters, indicating that templates capture dataset-specific reasoning patterns aligned with the semantic structure of queries. Notably, the legal-domain dataset (Housing QA) appears as a clearly distinct cluster, with its templates also separated from others. This separation suggests that templates not only reflect domain-specific reasoning structures, but also facilitate tight coupling between queries and reusable reasoning routines.

Usage Distribution and Co-occurrence. Figure 8 visualizes the distribution of template usage (Figure 12 with all datasets). We observe a pronounced long-tail pattern: a small number of templates are reused frequently across many queries, while the majority are invoked only occasionally. This distribution reveals the coexistence of general-purpose and specialized reasoning flows. To better understand template interactions, we compute pairwise co-occurrence statistics using lift values in Figure 9 (Figure 13 with all datasets), where a lift greater than 1 indicates above-chance co-usage. Several pair of templates exhibit consistently high lift, implying that certain reasoning templates are frequently recombined as stable compositional units, reflecting recurring reasoning routines. While datasets such as MuSiQue, CRAG, and FanOutQA display numerous template pairs with moderate lift values, signifying flexible and diverse reasoning combinations, Housing QA shows a contrasting trend: only a few pairs exhibit extremely high lift, with most others near independence. This suggests that legal domain queries are based on more rigid and repetitive reasoning structures, forming domain-specific "template bundles" rather than varied compositional patterns. Moreover, analyzing the top 10 most frequently co-occurring templates from MuSiQue reveals that 7 out of 10 originate from different training queries rather than a single source. This indicates that highly reused templates capture *reusable* reasoning primitives that can be flexibly recombined across queries to handle new questions. We also provide the example where two templates co-occurred in three different queries in Figure 14.

4.8 Qualitative Study

We present a case study example in Table 7. Given the query *"Why did Roncalli leave the place where Crucifixion's creator died?"*, both CiC and ToTAL had access to the same document set—specifically documents 359 and 228. CiC correctly identifies Titian as the creator and notes Roncalli's departure from Venice but fails to connect these pieces of information, concluding that it is unanswerable. In contrast, ToTAL leverages templates to decompose the reasoning process into explicit, interpretable steps: (1) attributing the artwork to its creator, (2) locating the creator's biographical and geographical context, and (3) linking these facts to the corresponding historical event. This structured reasoning chain enables the model to infer the answer correctly using the same evidence of CiC, highlighting how *thought templates provide the missing connective reasoning that bridges retrieved facts* into a coherent multi-hop explanation. Another example illustrating the effectiveness of the template update strategy is given in Appendix A.3.

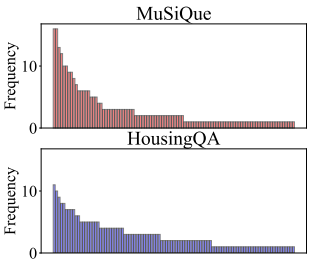

Figure 8: Template frequencies.

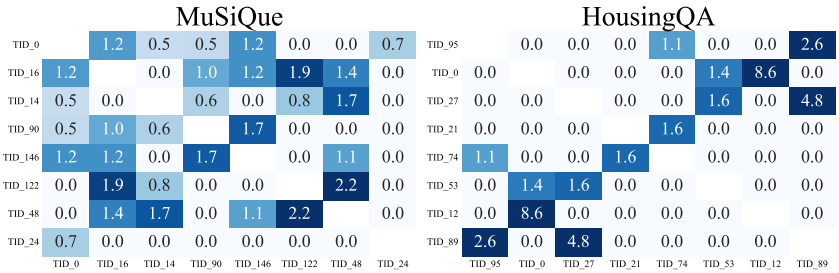

Figure 9: Template co-occurrence heatmap of lift values for MuSiQue and Housing QA.

5 Related Work

Long-Context Language Models Along with recent advances in LLMs, there has been substantial progress in extending their input capacity, now reaching hundreds of thousands or even millions of tokens (OpenAI, 2025a; Anthropic, 2025; Comanici et al., 2025) through architectural advances mechanisms (Su et al., 2021; Beltagy et al., 2020; Zaheer et al., 2020; Gu and Dao, 2024). This enables paradigm shifts such as placing all retrieved evidence into a single prompt, scaling many-shot in-context learning to unprecedented sizes (Lee et al., 2025; Baek et al., 2025; Chen et al., 2025), with recent benchmarks further probing these capabilities (Li et al., 2023; Zhang et al., 2024; Yang et al., 2025c; Bai et al., 2025; Lee et al., 2025).

Reasoning with Thought Templates Reasoning has been a central focus in improving the capabilities of LMs. Specifically, Chain-of-Thought prompting (Wei et al., 2022; Kojima et al., 2022) demonstrated that explicitly eliciting intermediate reasoning steps can largely enhance model performance, and diverse variants of it have been explored (Wang et al., 2023; Zhou et al., 2023; Kong et al., 2024; Aytes et al., 2025). Building on this, recent research explores augmenting LMs with structured reasoning patterns, often referred to as thought templates. For example, Yang et al. (2024b) proposed a framework that stores reasoning traces for math problem solving in a separate buffer and retrieves them when tackling new problems. Yang et al. (2025a) extended it to hierarchical reasoning by identifying optimal template paths via reinforcement learning. Similarly, Wu et al. (2024) employed Monte Carlo Tree Search to explore reasoning trajectories, while Yang et al. (2025b) distilled such strategies into smaller models with direct preference optimization. However, these approaches remain restricted to narrow domains (e.g., math) or rely on relatively simple reasoning steps without incorporating factual knowledge. Moreover, they typically retrieve a single template from an external buffer, whereas ours enables LCLMs to compose multiple templates simultaneously within a single generation process.

Text Gradient As directly updating or accessing the parameters of recent LMs is largely infeasible, a line of recent work has introduced natural-language feedback as a surrogate for gradients, effectively treating the LM itself as an optimizer (Pryzant et al., 2023; Yang et al., 2024a; Yüksekgönül et al., 2024; Cui et al., 2025). Whereas prior approaches primarily refine task or system prompts based on such feedback, our work instead updates reusable reasoning patterns by treating thought templates as learnable units refined through textual gradients.

6 Conclusion

We have presented ToTAL, a novel framework that fully leverages the capabilities of LCLMs by incorporating thought templates. ToTAL enables LCLMs to go beyond passive evidence consumption by combining factual documents with reusable reasoning patterns, and further refines these patterns through textual gradients without modifying model parameters. ToTAL consistently outperforms standard prompting and RAG baselines on diverse knowledge-intensive multi-hop reasoning benchmarks, across both idealized settings without retrieval and practical scenarios with retrieval, demonstrating the effectiveness of structured reasoning guidance within long-context settings. Our analyses reveal that thought templates not only improve factual accuracy but also exhibit meaningful compositionality, transferability across models, and domain-awareness. These findings collectively highlight a promising direction for augmenting LCLMs with reusable reasoning scaffolds, transforming them from passive knowledge consumers into strategy-driven reasoners.

Limitations

It is worth noting that our method achieves clear gains by combining compositional thought template design with iterative feedback-based refinement for complex knowledge-intensive tasks. Nevertheless, it assumes the availability of training queries and answers for template construction. In low-resource domains, this requirement may not be easily satisfied, and possible solutions include bootstrapping techniques or synthetic data generation. Second, within our template update framework, the feedback is produced by an auxiliary language model, which can be biased or noisy, potentially leading to suboptimal refinement; thus, exploring mitigation strategies could be an interesting direction for future work. Finally, while our current design focuses on textual templates, extending the framework to more structured templates or multimodal contexts could further broaden its applicability.

Ethics Statement

As our approach feeds LCLMs with a large amount of evidence documents (or sometimes the entire corpus), there is a possibility that some of these may contain harmful, sensitive, or personally identifiable information. We recommend that practitioners remain mindful of such risks and consider incorporating bias detection and mitigation strategies when deploying our method.

References

Rishabh Agarwal, Avi Singh, Lei M. Zhang, Bernd Bohnet, Stephanie Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, John D. Co-Reyes, Eric Chu, Feryal M. P. Behbahani, Aleksandra Faust, and Hugo Larochelle. 2024. Many-shot in-context learning. *ArXiv*, abs/2404.11018.

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sébastien Bubeck, Che Chang, Kai Chen, and 105 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, abs/2508.10925.

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, and 33 others. 2023. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Anthropic. 2025. claude.

Simon A. Aytes, Jinheon Baek, and Sung Ju Hwang. 2025. Sketch-of-thought: Efficient LLM reasoning with adaptive cognitive-inspired sketching. *arXiv preprint arXiv:2503.05179*, abs/2503.05179.

Jinheon Baek, Sun Jae Lee, Prakhar Gupta, Geunseob Oh, Siddharth Dalmia, and Prateek Kolhar. 2025. Revisiting in-context learning with long context language models. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 26950–26966. Association for Computational Linguistics.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2025. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 3639–3664. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, abs/2004.05150.

Zihan Chen, Song Wang, Zhen Tan, Xingbo Fu, Zhenyu Lei, Peng Wang, Huan Liu, Cong Shen, and Jundong Li. 2025. A survey of scaling in large language model reasoning. *ArXiv*, abs/2504.02181.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit S. Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 81 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, abs/2507.06261.

Wendi Cui, Jiaxin Zhang, Zhuohang Li, Hao Sun, Damien Lopez, Kamalika Das, Bradley Malin, and Kumar Sricharan. 2025. A survey of automatic prompt optimization with instruction-focused heuristic-based search algorithm.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. volume abs/2312.00752.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 180 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Nature*, 645:633–638.

Wenyu Huang, Pavlos Vougiouklis, Mirella Lapata, and Jeff Z. Pan. 2025. Masking in multi-hop QA: an analysis of how language models perform with context permutation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 17781–17795. Association for Computational Linguistics.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 7036–7050. Association for Computational Linguistics.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, Xin Zhou, Enzhi Wang, and Xiaohang Dong. 2024. Better zero-shot reasoning with role-play prompting. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 4099–4113. Association for Computational Linguistics.

Jinhyuk Lee, Anthony Chen, Zhuyun Dai, Dheeru Dua, Devendra Singh Sachan, Michael Boratko, Yi Luan, Sébastien M. R. Arnold, Vincent Perot, Siddharth Dalmia, Hexiang Hu, Xudong Lin, Panupong Pasupat, Aida Amini, Jeremy R. Cole, Sebastian Riedel, Iftekhar Naim, Ming-Wei Chang, and Kelvin Guu. 2025. LOFT: scalable and more realistic long-context evaluation. In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 6698–6723. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023. Loogle: Can long-context language models understand long contexts? *ArXiv*, abs/2311.04939.

OpenAI. 2025a. Gpt-5 system card.

OpenAI. 2025b. Introducing gpt-4.1 in the api.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7957–7968. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*, volume 500-225 of *NIST Special Publication*, pages 109–126. National Institute of Standards and Technology (NIST).

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *ArXiv*, abs/2104.09864.

Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *ArXiv*, abs/2401.15391.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *ACL*.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 2609–2634. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Jinyang Wu, Mingkuan Feng, Shuai Zhang, Feihu Che, Zengqi Wen, and Jianhua Tao. 2024. Beyond examples: High-level automated reasoning paradigm in in-context learning via MCTS. *arXiv preprint arXiv:2411.18478*, abs/2411.18478.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2024a. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ling Yang, Zhaochen Yu, Bin Cui, and Mengdi Wang. 2025a. Reasonflux: Hierarchical LLM reasoning via scaling thought templates. *arXiv preprint arXiv:2502.06772*, abs/2502.06772.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. 2024b. Buffer of thoughts: Thought-augmented reasoning with large language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Minkai Xu, Joseph E. Gonzalez, Bin Cui, and Shuicheng Yan. 2025b. Supercorrect: Advancing small LLM reasoning with thought template distillation and self-correction. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Van Yang, Hongye Jin, Shaochen Zhong, Song Jiang, Qifan Wang, Vipin Chaudhary, and Xiaotian Han. 2025c. 100-longbench: Are de facto long-context benchmarks literally evaluating long-context ability? In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 17560–17576. Association for Computational Linguistics.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, and 8 others. 2024c. CRAG - comprehensive RAG benchmark. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Mert Yüksekgönül, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. 2024. Textgrad: Automatic "differentiation" via text. *arXiv preprint arXiv:2406.07496*, abs/2406.07496.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024. ınftybench: Extending long context evaluation beyond 100k tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 15262–15277. Association for Computational Linguistics.

Lucia Zheng, Neel Guha, Javokhir Arifov, Sarah Zhang, Michal Skreta, Christopher D. Manning, Peter Henderson, and Daniel E. Ho. 2025. A reasoning-focused legal retrieval benchmark. In *Proceedings of the 2025 Symposium on Computer Science and Law, CSLAW 2025, Munich, Germany, March 25-27, 2025*, pages 169–193. ACM.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Andrew Zhu, Alyssa Hwang, Liam Dugan, and Chris Callison-Burch. 2024. Fanoutqa: A multi-hop, multi-document question answering benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics, ACL 2024 - Short Papers, Bangkok, Thailand, August 11-16, 2024*, pages 18–37. Association for Computational Linguistics.

Table 6: Iteration-wise decision summary of template updates.

		KEEP	ADD	FIX	DISCARD	F1
MuSiQue	CiC	-	-	-	-	63.87
	Iter. 0	-	-	-	-	70.51
	Iter. 1	4	0	10	0	71.39
	Iter. 2	2	1	9	0	73.30
	Iter. 3	1	0	7	0	71.07
CRAG	CiC	-	-	-	-	17.32
	Iter. 0	-	-	-	-	27.60
	Iter. 1	1	0	14	0	28.61
	Iter. 2	0	0	14	2	30.08
	Iter. 3	1	0	15	0	25.55

A Experimental Results

A.1 Template Statistics

We generate thought templates from 50 question–answer pairs, using a detailed prompt shown in Figure 15. The initial template pool consists of 172 templates for MuSiQue, 162 for CRAG, 133 for FanOutQA, and 149 for HousingQA. These templates are then iteratively updated, as described in the following subsection.

A.2 Template Update Strategy

To refine the initial pool of reasoning templates, we adopt an iterative update strategy guided by textual-gradient signals. At each iteration, candidate templates are categorized into four actions: `KEEP`, `ADD`, `FIX`, or `DISCARD`.

Table 6 summarizes the update dynamics on MuSiQue and CRAG. In early iterations, the majority of updates correspond to `FIX`. As iterations proceed, the ratio of `KEEP` and `ADD` increases, indicating that high-quality templates become more stable and occasionally expand with new variants.

This dynamic is also reflected in performance: on MuSiQue, F1 steadily improves up to the second iteration but slightly declines afterward, while on CRAG, moderate gains appear until the second iteration before over-refinement in the third iteration leads to a drop. These results suggest that the update process converges, with templates stabilizing into reusable reasoning patterns and only limited gains beyond a few iterations, resembling the diminishing returns observed in typical ML training.

A.3 Qualitative Study on Template Update

We further investigate the effectiveness of template update qualitatively. As shown in Figure 10, the original TID_91 was decided to be `FIX` because feedback revealed that it often broke the reasoning chain in multi-hop settings. In particular, it failed to properly integrate the output of previous steps (e.g., resolving a township into its containing county) and gave only vague instructions such as "select the relevant adjacent territory based on context," without specifying how to apply additional constraints like naming requirements. After being updated, the revised TID_91' addressed these weaknesses by explicitly incorporating the reference territory from upstream steps, adding a filtering stage to enforce contextual conditions, and validating that the final selected territory satisfied all requirements.

Table 8 illustrates the effect of this refinement. With the original TID_91, the model incorrectly answered "Oconto Falls," as it failed to maintain continuity with the birthplace entity and did not filter adjacent candidates. In contrast, the updated TID_91' produced the correct answer "Green Bay," demonstrating that the refined template successfully chained outputs across steps and enforced query-specific constraints. This case highlights that our template update strategy makes the reasoning process more consistent and reliable in multi-hop queries.

Table 7: Case study on the multi-hop query from MuSiQue.

Query	Why did Roncalli leave the place where Crucifixion's creator died?
Gold Answer	for the conclave in Rome
CIC	Looking at the documents to trace this query: TITLE: Crucifixion (Titian) | ID: 359 TITLE: Pope John XXIII | ID: 228 From document ID: 359, I can see that the Crucifixion was created by Titian, a Venetian artist, and the painting is located in Ancona. However, this document doesn't specify where Titian died. From document ID: 228, I can see that Roncalli (who became Pope John XXIII) left Venice for the conclave in Rome, and it mentions he had a return train ticket to Venice, suggesting he expected to return but didn't because he was elected Pope. However, the documents don't explicitly state where Titian died or provide a clear connection between Titian's death location and Roncalli's departure from that place. The query cannot be fully answered with the information provided in these documents. Final Answer: ['Cannot be determined from the provided documents']
TOTAL (Ours)	Looking at this query, I need to identify who created "Crucifixion," where they died, and why Roncalli left that place. Step 1 | TEMPLATE_TITLE: Work-to-Creator Attribution TEMPLATE_ID: TID_77 | TEMPLATE_CONTENT: Descriptions: Finding the creator or contributor associated with a specific creative work such as films, books, or other media. Reason Flow: ['Identify the specific creative work in question', 'Determine what type of contributor information is needed (screenwriter, director, etc.)', 'Look up the creative credits or attribution for that work', "Extract the specific person's name who fulfilled that role"] Example: 'example_problem': "Who was the screenwriter of 'With the Mounted Police'?", 'solution_steps': ["Identify the film: 'With the Mounted Police'", 'Look up the screenwriting credits for this film', 'Find that Lloyd Lonergan was the screenwriter'], 'final_answer': 'Lloyd Lonergan' Step 2 | TEMPLATE_TITLE: Biographical Location Lookup TEMPLATE_ID: TID_58 | TEMPLATE_CONTENT: Descriptions: Finding the specific location associated with a significant life event of a person, such as where they died, were born, or lived. Reason Flow: ['Identify the specific person in question', 'Determine what type of location information is needed (death, birth, residence, etc.)', 'Research biographical records or databases', 'Extract the specific location associated with that life event'] Example: 'example_problem': 'In what city did Lloyd Lonergan die?', 'solution_steps': ['Identify the person: Lloyd Lonergan', 'Research biographical information about his death', 'Find records indicating he died in New York'], 'final_answer': 'New York' Step 3 | TEMPLATE_TITLE: Historical Event Specification TEMPLATE_ID: TID_139 | TEMPLATE_CONTENT: Descriptions: Determining specific historical events or circumstances related to a known figure, particularly focusing on significant life events. Reason Flow: ['Focus on the specific aspect being questioned (e.g., cause of death, major events)', 'Research the historical record for that specific detail', 'Identify the commonly accepted historical fact'] Example: 'example_problem': 'What was the cause of death of Jesus?', 'solution_steps': ['Focus on the specific question: cause of death', 'Research historical accounts of Jesus's death', 'Identify the commonly accepted cause: crucifixion'], 'final_answer': 'crucifixion' TITLE: Crucifixion (Titian) | ID: 359 TITLE: Pope John XXIII | ID: 228 Final Answer: ['for the conclave in Rome']

TID_91 — Adjacent Territory Identification (Original)

Description. Finding administrative territories that share borders or are adjacent to a given territory.

Reason flow.

1. Identify the reference territory
2. Search for border or adjacency relationships
3. List all territories that share borders
4. Select the relevant adjacent territory based on context

Example.

- **Problem:** Which county shares a border with Lincoln County?
- **Solution steps:**
  - Identify the reference territory: Lincoln County
  - Search for counties that share borders with Lincoln County
  - Identify Nye County as one that shares a border with Lincoln County
- **Final answer:** `Nye County`

TID_91' — Adjacent Territory Identification (Revised)

Description. Finding administrative territories that share borders or are adjacent to a given territory, including cases where the reference territory must first be determined from contained entities.

Reason flow.

1. Identify or receive the reference territory from previous steps
2. If reference territory contains sub-entities, confirm the containing territory
3. Search for all territories that share borders with the reference territory
4. Apply additional filtering criteria from the query context
5. Validate that selected adjacent territory meets all constraints
6. Select the final adjacent territory that matches all requirements

Example.

- **Problem:** Which county shares a border with Dearborn County and is named after a river?
- **Solution steps:**
  - Identify the reference territory: Dearborn County
  - Search for all counties that share borders with Dearborn County
  - List adjacent counties: Ohio, Ripley, Franklin
  - Apply filtering: must be named after a river
  - Validate: Ohio County is named after Ohio River
  - Select Ohio County
- **Final answer:** `Ohio County`

Figure 10: Comparison of **TID_91** before and after refinement.

Feedback for TID_91 (from textual-gradient signals).

- Failed to chain with previous step (Miller Township → Dearborn County).
- Jumped to irrelevant result (*Río de la Plata*) instead of adjacent counties.
- Missing handling of multi-step adjacency (contained entities).
- Vague filtering step: *"Select based on context"*.
- Needs explicit integration with prior outputs.
- Missing validation against query constraints (e.g., named after a river).

Figure 11: Textual-gradient feedback guiding the refinement of **TID_91**.

Table 8: Case study on showing the effect of refining **TID_91**.

Query	What city is near the birthplace of Erik Jensen (American football) and shares a border with that place?
Gold Answer	Green Bay
Before (TID_91)	Step 1 | TEMPLATE_TITLE: Biographical Location Lookup TEMPLATE_ID: TID_48 TEMPLATE_CONTENT: Descriptions: Finding specific geographic locations associated with important life events of a person ... Step 2 | TEMPLATE_TITLE: Administrative Capital Lookup TEMPLATE_ID: TID_145 TEMPLATE_CONTENT: Descriptions: Finding the capital city or administrative center of a given political entity ... Step 3 | TEMPLATE_TITLE: Adjacent Territory Identification TEMPLATE_ID: **TID_91** TEMPLATE_CONTENT: Descriptions: Finding administrative territories that share borders or are adjacent to a given territory ... Step 4 | TEMPLATE_TITLE: Administrative Capital Lookup TEMPLATE_ID: TID_145 TEMPLATE_CONTENT: Descriptions: ... TITLE: Erik Jensen (American football) | ID: 367 TITLE: Pulaski High School | ID: 26 TITLE: Kelly Brook, Wisconsin | ID: 371 Final Answer: ['Oconto Falls']
After (TID_91')	Step 1 | TEMPLATE_TITLE: Person-to-Birthplace Mapping TEMPLATE_ID: TID_105 TEMPLATE_CONTENT: Descriptions: Finding the birthplace or place of origin of a specific person ... Step 2 | TEMPLATE_TITLE: Adjacent Territory Identification TEMPLATE_ID: **TID_91' (Revised)** TEMPLATE_CONTENT: Descriptions: Includes confirmation of containing territories and applying filters ... Step 3 | TEMPLATE_TITLE: Administrative Capital Lookup TEMPLATE_ID: TID_145 TEMPLATE_CONTENT: Descriptions: Finding the capital city or administrative center of a given political entity ... TITLE: Erik Jensen (American football) | ID: 367 TITLE: Pulaski High School | ID: 26 Final Answer: ['Green Bay']

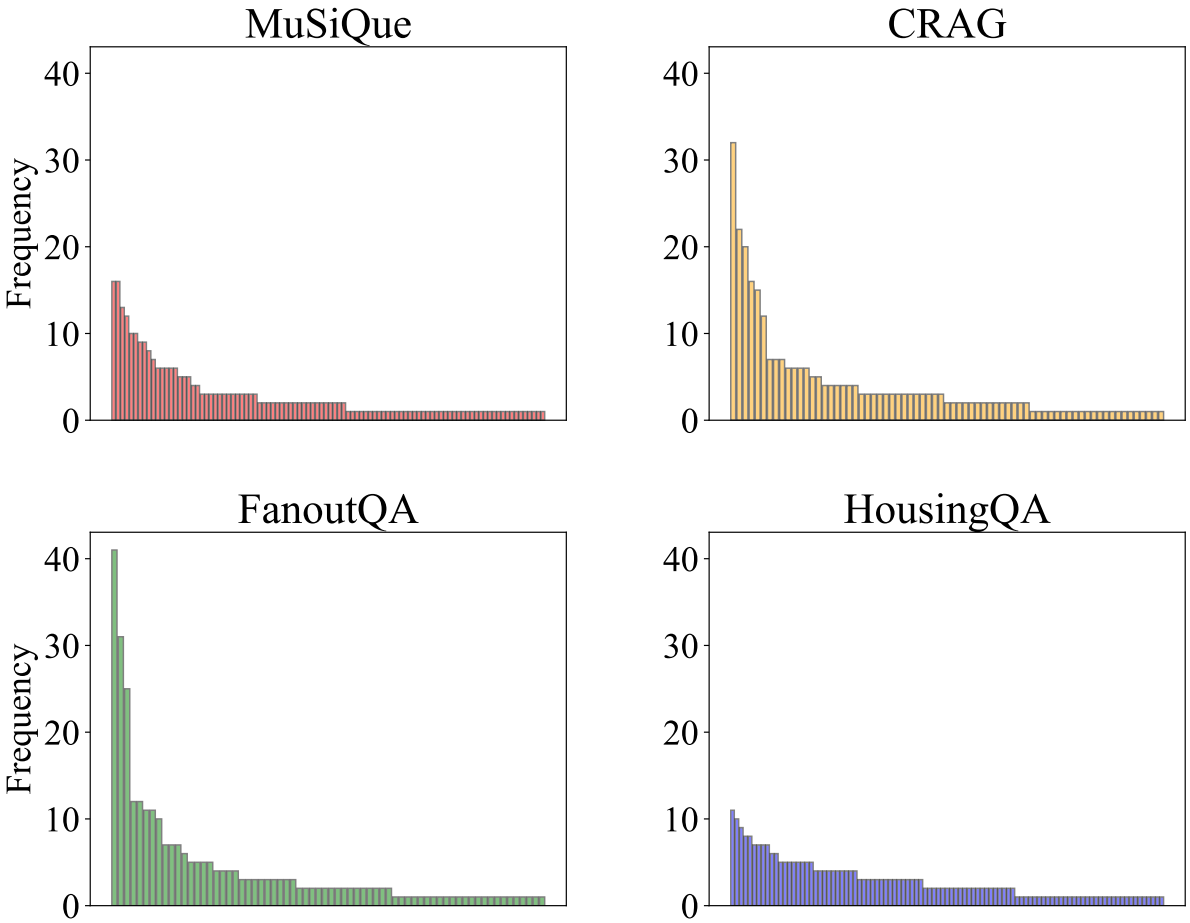

Figure 12: Histogram of template frequencies across datasets.

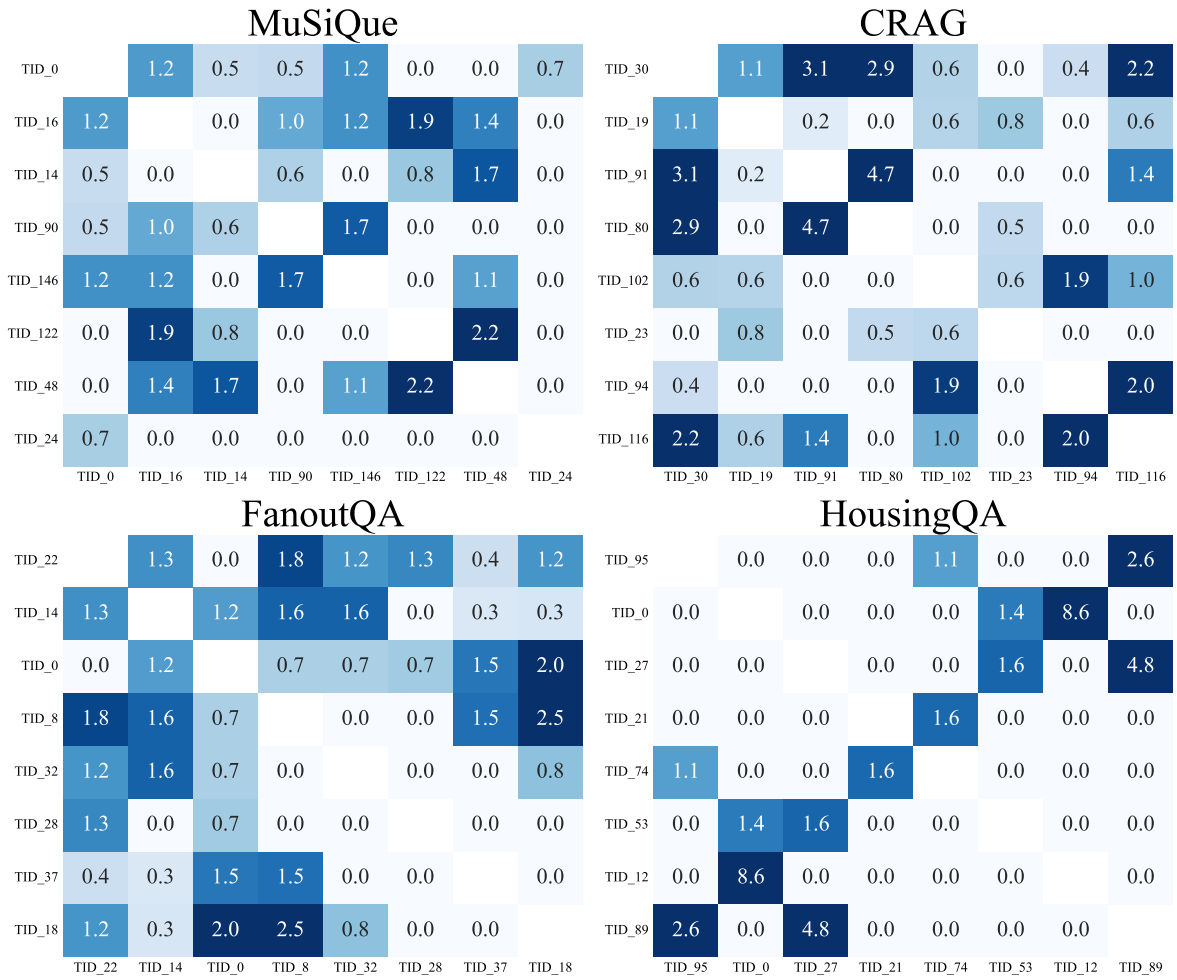

Figure 13: Template co-occurrence heatmap of lift values across datasets.

TID_45 — Administrative Territory Identification

Description. A method for determining which administrative territorial entity (state, province, country) contains a specific location.

Reason flow.

1. Identify the specific location
2. Determine the type of administrative division needed (state, province, country, etc.)
3. Identify which administrative entity contains that location

Example.

- **Problem:** What state is Boston located in?
- **Solution steps:**
  - Identify the location: Boston
  - Determine the administrative level needed: state level
  - Identify the containing administrative entity: Massachusetts
- **Final answer:** `Massachusetts`

TID_65 — Demographic Ranking and Selection

Description. Finding the top-ranked entity within a geographic region based on specific demographic criteria.

Reason flow.

1. Identify the geographic scope for comparison
2. Determine the ranking criteria (population, area, etc.)
3. Apply the criteria to find the top-ranked entity
4. Verify the result meets all specified conditions

Example.

- **Problem:** What city is Russia's largest metropolitan area as measured by population?
- **Solution steps:**
  - Identify that we need the largest metropolitan area in Russia
  - Apply population-based ranking criteria
  - Determine that Moscow has the largest metropolitan population in Russia
- **Final answer:** `Moscow`

Queries using these templates (chains only).

Query 1. Who won the Indy Car Race in the largest populated city of the state where the performer of *Mingus Three* is from?
Templates applied: TID_14 → TID_105 → TID_45 → TID_65 **Final Answer:** `Mario Andretti`

Query 2. What was the wettest year in the second largest city in the state where Yuma's Library District is located?
Templates applied: TID_45 → TID_65 → TID_90 **Final Answer:** `1905`

Query 3. How long are the city council terms in the second largest city in the state where Yuma is located?
Templates applied: TID_45 → TID_65 → TID_66 **Final Answer:** `four-year terms`

Figure 14: Two templates (**TID_45**, **TID_65**) and three queries that used these templates simultaneously.

Template Construction Prompt.
You are an expert in reasoning strategies. Given a complex, multi-step problem, its complete solution, and the final answer, extract a structured problem-solving template composed of reusable sub-templates. Return the result in JSON format with the following structure:

1. A clear name for the strategy (`template_name`)
2. A brief description of the method (`description`)
3. A step-by-step reasoning flow to solve similar problems (`reason_flow`)
4. An example application, including:
   - Problem statement (`example_problem`)
   - Solution steps (`solution_steps`)
   - Final answer (`final_answer`)
5. `sub_templates`: A list of dictionaries, each representing a reasoning sub-template with:
   - `template_name`: A descriptive name for this sub-strategy
   - `description`: A brief description of the sub-strategy
   - `reason_flow`: A list of reasoning steps involved in this sub-task
   - `example`: An example application of this sub-template, including:
     - `example_problem`: A question matching this reasoning pattern
     - `solution_steps`: Step-by-step solution to that question
     - `final_answer`: The answer to that question

Instruction constraint: Respond only in JSON format with no explanation.

Inputs shown to the model:

```
Problem:
""" {problem} """

Solution:
""" {solution} """

Final Answer:
""" {answer} """
```

Figure 15: Prompt used to construct compositional thought templates from (Problem, Solution, Final Answer). Each generated sub-template is treated as a template and added to $\mathcal{T}$.

Role. You are improving a reasoning template where it was applied.
Current Template.

```
{JSON dump of the current template:
 "template_id","template_name","description","reason_flow","example"}
```

Failed Cases where this template was used.

```
Case #0 (F1: {f1})
Query: {query text}
REASONING TRACE: {model_outputs[0]}
Gold: {gold answer}
Pred: {prediction}
```

Failed Case Source (original query/solution/answer).

```
Query: {problem}
Solution Steps:
{step-by-step solution or evidence block}
Final Answer: {answer}
```

Your task. Analyze the template's role in the prediction error:

- How the template led to the incorrect prediction
- What needs to be fixed in the template
- Specific feedback to get the correct answer

Decision Guide (choose exactly one at the end).

- **FIX** – Template needs revision to address the issues above
- **DISCARD** – Template is fundamentally incorrect
- **KEEP** – Template works perfectly AND failure is due to external factors (e.g., answer format)
- **ADD** – Template works perfectly BUT failure is due to system coordination issues (e.g., selection, multi-step integration)

Output format.

- Return bullets only for your analysis.
- On the **FINAL LINE**, output **exactly one** of: `**FIX**` or `**DISCARD**` or `**ADD**` or `**KEEP**`.

Figure 16: Prompt for generating textual gradient feedback.

Role. You will edit a reasoning template based on the FEEDBACK.
Output constraints.

- Return **ONLY** a valid JSON object matching the SCHEMA below.
- **No markdown, no extra text.** Use double quotes for all keys/strings.

SCHEMA.

```
{
  "template_id": "string",
  "template_name": "string",
  "description": "string",
  "reason_flow": ["string", "..."],
  "example": {
    "example_problem": "string",
    "solution_steps": ["string", "..."],
    "final_answer": "string"
  }
}
```

Current Template.

```
{JSON dump of the current template:
 "template_id","template_name","description","reason_flow","example"}
```

Failed Cases (referenced in feedback).

```
Case #0 (F1: {f1})
Query: {query text}
REASONING TRACE: {model_outputs[0]}
Gold: {gold answer}
Pred: {prediction}
```

Failed Case Source (original query/solution/answer).

```
Query: {problem}
Solution Steps:
{step-by-step solution or evidence block}
Final Answer: {answer}
```

FEEDBACK. (from Fig. 16)

```
{feedback text}
```

Instruction. Revise the template to address the FEEDBACK while preserving reusable structure and staying within the SCHEMA. **Respond only with the JSON object.**

Figure 17: Prompt for template update given textual gradient feedback.