Decoupled Reasoning with Implicit Fact Tokens (DRIFT): A Dual-Model Framework for Efficient Long-Context Inference

Anonymous ACL submission

Abstract

The integration of extensive, dynamic knowledge into Large Language Models (LLMs) remains a significant challenge due to the inherent entanglement of factual data and reasoning patterns. Existing solutions, ranging from non-parametric Retrieval-Augmented Generation (RAG) to parametric knowledge editing, are often constrained in practice by finite context windows, retriever noise, or the risk of catastrophic forgetting. In this paper, we propose **DRIFT**, a novel dual-model architecture designed to explicitly decouple knowledge extraction from the reasoning process. Unlike static prompt compression, DRIFT employs a lightweight knowledge model to dynamically compress document chunks into implicit fact tokens conditioned on the query. These dense representations are projected into the reasoning model's embedding space, replacing raw, redundant text while maintaining inference accuracy. Extensive experiments show that DRIFT significantly improves performance on **long-context tasks**, outperforming strong baselines among comparably sized models. Our approach provides a scalable and efficient paradigm for extending the effective context window and reasoning capabilities of LLMs. Our code and data will be made public upon publication.

1 Introduction

The applicability of Large Language Models (LLMs) to knowledge-intensive tasks is limited by the static nature of their pre-training data. To address this limitation, prior work has explored two complementary strategies.The first focuses on augmenting the input context, while the second emphasizes knowledge parameter editing.

Traditional input augmentation via RAG or long-context prompting is increasingly constrained by the "retriever's ceiling" and the quadratic computational costs of processing long sequences. Neural compression methods (e.g., COCOM (Rau et al., 2024), C3 (Liu and Qiu, 2025)) attempt to mitigate this by distilling text into latent representations; however, as they primarily focus on static compression, task-critical information relevant to the query is frequently lost. Conversely, internalizing knowledge through direct parametric updates, such as fine-tuning or knowledge editing, often disrupts the inherent coupling between a model's internal knowledge and its reasoning logic, while also risking catastrophic forgetting. While modular approaches like MLP Memory (Wei et al., 2025) offer a plug-and-play alternative, they remain bound to pre-indexed resources and struggle to handle instantaneous, unseen long-context inputs in real-time.

To address these challenges, we introduce **DRIFT**, a dual-model architecture that decouples context processing from core reasoning. In this framework, a lightweight **knowledge model** extracts query-relevant information from document chunks and compresses it into high-density **implicit fact tokens** within a latent space. These tokens serve as concise, knowledge-rich representations that are projected into a larger **reasoning model's** embedding space. The reasoning model can perform sophisticated inference efficiently based on this compact factual context instead of raw text, even in long-context or knowledge-intensive scenarios. By delegating the processing of redundant background knowledge to the knowledge module, this design allows the reasoning model to remain unburdened by raw context, focusing instead on "clean" and deep inference through the distilled factual representations.

Our main contributions are as follows:

- **A Decoupled Inference Paradigm for Large Language Models:** We propose a dual-model framework that decouples knowledge extraction from reasoning. A lightweight knowledge model encodes query-relevant informa-

tion into compact fact tokens, which are consumed by a larger reasoning model for inference. Compared with directly processing long contexts, DRIFT improves performance while substantially reducing inference latency, achieving **an average 7× speedup on 256k-token documents**.

- **Expanding Effective Context Window with High-Ratio Compression:** By encoding extensive textual knowledge into compact fact tokens, our framework significantly extends the model's usable context window. Specifically, our **DRIFT** model (based on Mistral 7B) achieves a **32× compression ratio** while **improving accuracy from 20.87% to 27.54%** on the LongBench v2 benchmark, demonstrating superior reasoning capabilities with substantially reduced time overhead. Furthermore, even under more aggressive compression settings (64× and 128×), DRIFT remains highly competitive, indicating strong robustness to extreme compression ratios.

- **Comprehensive Empirical Analysis and Resource Contribution:** We construct a large-scale Document–QA–Evidence dataset with fine-grained supervision, comprising over **300K** instances and documents ranging from **1K** to **8K** tokens. We further conduct extensive ablation studies and quantitative evaluations, rigorously validating the framework and demonstrating its robustness.

2 Related Work

2.1 Prompt Compression

Directly feeding new knowledge as context into Large Language Models (LLMs) is constrained by limited context windows and incurs significant overhead in both memory and computation. To mitigate these issues, various prompt compression methods have been proposed, which generally fall into two paradigms: *Hard Compression* and *Soft Compression*.

Hard Compression (Token Selection). Hard compression methods, also known as token pruning or selection, aim to reduce input length by discarding tokens deemed less informative. Early approaches like Selective Context (Li et al., 2023) utilize self-information or perplexity metrics to filter out redundancy. More advanced frameworks, such as LongLLMLingua (Jiang et al., 2024) and LLMLingua-2 (Pan et al., 2024), perform prompt compression via task-aware coarse-to-fine filtering and distillation-based token selection, respectively. Despite these advancements, hard compression approaches inherently limit the model's reasoning potential. They **irreversibly discard information** and rely on rigid, locally made retention decisions that often fail to preserve the **global semantic structure** required for reliable complex reasoning.

Soft Compression (Latent Representation). Early soft compression approaches, such as AutoCompressor, Gist Tokens, and ICAE, integrate compression and reasoning within a single language model. While effective for moderate context reduction, this tightly coupled design makes it difficult for a single model to simultaneously excel at both high-fidelity compression and complex reasoning, particularly under extreme compression ratios. To address this limitation, subsequent work explores decoupling compression from reasoning. Methods such as xRAG (Cheng et al., 2024) and COCOM (Rau et al., 2024) build upon the Retrieval-Augmented Generation (RAG) paradigm by compressing retrieved documents into compact latent representations before passing them to the language model. Although effective in reducing input length, these approaches remain fundamentally constrained by the retrieval stage and inherit the upper-bound limitations of RAG systems. Beyond RAG-based compression, E2LLM (Liao et al., 2025) employs a lightweight encoder to compress long inputs into latent representations for downstream LLM reasoning, though the model weights are not publicly released, limiting reproducibility and practical adoption. Context Cascade Compression (C3) (Liu and Qiu, 2025) demonstrates strong compression fidelity but lacks task-specific adaptation for downstream reasoning. As a result, a gap remains between compressed representation understanding and effective reasoning.

However, most existing soft compression methods operate in a **static** and query-agnostic manner, producing generic compressed representations that often fail to preserve task-critical information under high compression ratios. In contrast, **DRIFT** adopts a **dynamic, query-conditioned compression strategy** that selectively encodes relevant information, enabling effective reasoning even under **extreme compression ratios** in knowledge-intensive scenarios.

2.2 Learned Memory

nother line of work introduces learned parametric memory modules, such as Memory Decoder (Cao et al., 2025) and MLP Memory (Wei et al., 2025), which store knowledge in trainable parameters and are often pretrained to emulate retrieval behavior. These methods reduce inference latency without modifying the underlying model parameters, thereby preserving general reasoning capabilities. However, these architectures are inherently static-resource bound: they rely on pre-indexing or offline training on fixed knowledge bases, rendering them incapable of handling instantaneous, long-context inputs in real-time. Moreover, as these modules are often pre-trained to emulate or compress retriever behaviors, their inference dynamics remain tethered to the limitations of the original retrieval paradigm.

3 Methodology

Our proposed strategy encourages the knowledge model to perform information-adaptive compression, rather than static compression. This compels the model to first identify and abstract core informational content before encoding it into a compressed semantic representation. As a result, the model learns to prioritize semantically meaningful content over superficial token patterns, leading to improved generalization and robustness in downstream tasks.

3.1 Bucketed Compression: Beyond Fixed-Ratio Compression

Most existing context compression methods adopt a fixed-ratio strategy, where the number of compressed tokens is strictly proportional to the input length (e.g., compressing 128 input tokens into 16 output tokens for an 8:1 ratio). However, such a design implicitly assumes that informative content is evenly distributed across the input, which rarely holds in real-world tasks. The amount of query-relevant information within a given context is always unknown and varies in token length. Certain tokens (such as numbers, named entities, main verbs, and constraint-related words) carry a large amount of task-relevant information, whereas redundant modifiers and generic, content-free sentences contribute little to understanding the context. For example, in document-grounded QA or long-form reasoning, a few critical sentences may carry the majority of the answer-relevant information.

This fixed-ratio compression can thus become fragile in scenarios where the input contains sparse but crucial evidence. Moreover, training a model to uniformly compress variable-length sequences may encourage shortcut learning (e.g., positional bias, over-averaging), hindering semantic abstraction. To address this, we propose a Bucketed Compression strategy that shifts from ratio-based compression to range-based compression. Instead of computing the number of output tokens as a fixed fraction of the input, we predefine token-length buckets (e.g., 64–128, 128–256), and map each input in a bucket to a fixed-size output based on the upper bound of that range.

To make the distinction clear, we present a direct comparison of the two strategies in formulaic form, under the assumption of a compression ratio c:

$$\xi_{\text{uniform}} = \left\lceil \frac{n}{c} \right\rceil, \qquad \xi_{\text{bucket}} = \left\lceil \frac{b(n)}{c} \right\rceil$$

where $b(n)$ denotes the upper bound of the bucket containing n tokens.

3.2 DRIFT: Decoupled Reasoning with Implicit Fact Tokens

The core idea of DRIFT is to modularize and explicitly separate knowledge reading and reasoning. Specifically, a small-scale knowledge model (ψ_{kno}) is responsible for reading long documents and compressing them into query-relevant information, while a large-scale reasoning model ψ_{kno} focuses on utilizing this compressed knowledge to perform complex reasoning and generate answers. The interaction between the two models is realized in the latent space, which reduces redundancy and mitigates the risk of irrelevant noise interfering with the reasoning process.

To facilitate a clear understanding of our method, the overall workflow of DRIFT is depicted in Figure 1. We first define a document as $X = (x_1, \ldots, x_n)$, where n is the number of tokens in the document. For long input contexts, a systematic document chunking strategy was employed within the DRIFT framework. We utilized the `RecursiveCharacterTextSplitter` from the LangChain framework (Contributors, 2024) for this purpose. This recursive approach was adopted to ensure optimal semantic coherence by prioritizing natural delimiters (e.g., paragraphs and sentences).

$$X \xrightarrow{\text{Split}} C = (C_1, C_2, \ldots, C_K). \tag{1}$$

Given a query Q, we append a fixed number of <|CPS|> tokens to each chunk C_j and process them

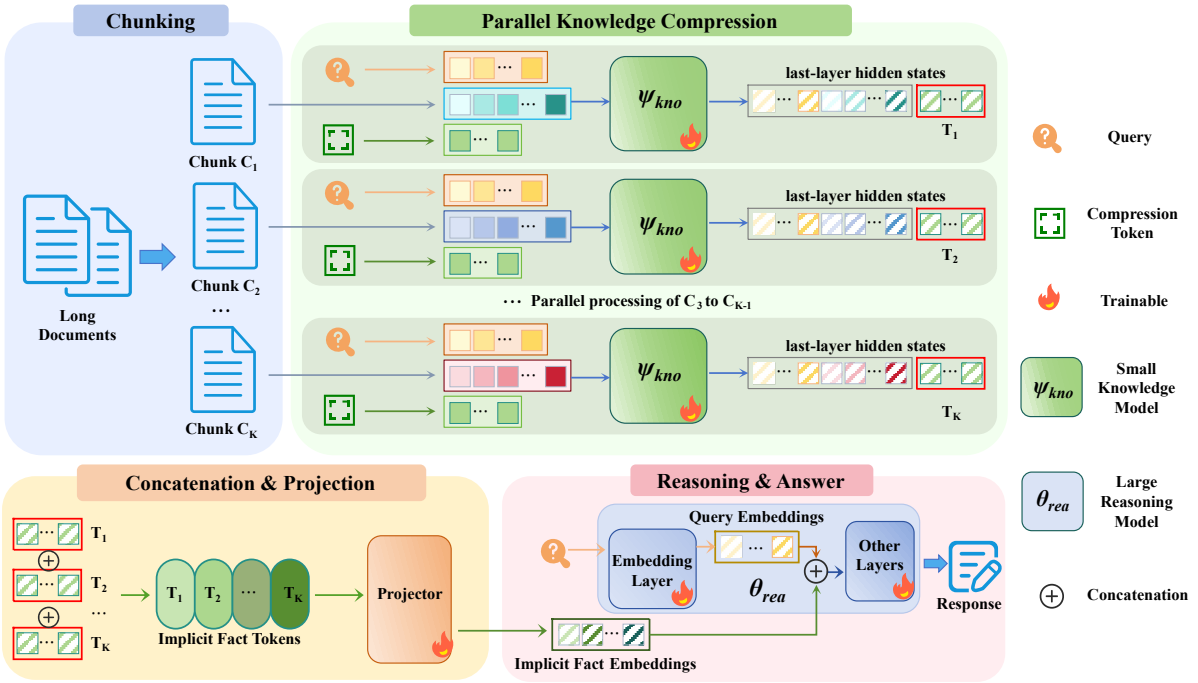

Figure 1: **The overall workflow of DRIFT.** DRIFT implements knowledge compression and decoupled reasoning in four steps. **Step 1:** The long document X is recursively partitioned into semantically coherent chunks to preserve structural integrity. **Step 2:** The small knowledge model ψ_{kno} compresses query-relevant information from each chunk in parallel into latent implicit fact tokens T_J. **Step 3:** The latent tokens are concatenated and mapped by an MLP projector π to align with the reasoning model's embedding space. **Step 4:** The large reasoning model θ_{rea} generates the final response by performing efficient inference on the concatenated embeddings.

in parallel with the knowledge model ψ_{kno}, using the last-layer hidden states of these compression tokens as the latent representation T_j.

$$\psi_{\text{kno}} : (C_j, Q) \rightarrow T_j \in \mathbb{R}^{\xi_j \times d}. \tag{2}$$

Finally, the outputs from all chunks are concatenated in the original order to yield the global sequence of implicit fact tokens, denoted T.

$$T = \text{Concat}(T_1, \ldots, T_K) = [t_1, \ldots, t_\xi] \in \mathbb{R}^{\xi \times d}, \tag{3}$$

where $\xi = \sum_{j=1}^{K} \xi_j \ll N$.

Implementation Note: During concatenation, a double newline separator (\n\n) is inserted between adjacent sub-sequences T_j and T_{j+1} to mark chunk boundaries. We define $T_i \in \mathbb{R}^d$ as implicit fact tokens, and d denotes the hidden state dimensionality. These tokens encapsulate essential information from the input and serve as continuous latent units for the reasoning model.

In the next step, a three-layer MLP projector π maps the implicit fact tokens into implicit fact embeddings $E = (e_1, e_2, \ldots, e_\xi)$, thereby aligning them with the embedding space of the reasoning model. These embeddings, together with the query embeddings $E(Q)$, are subsequently fed into the reasoning model for downstream inference.

$$\theta_{\text{rea}} : \text{Concat}(E, E(Q)) \rightarrow \text{Response} \tag{4}$$

The Response denotes the thoughts and the final answer generated by the reasoning model.

This approach not only substantially reduces GPU memory consumption but also frees the reasoning model from processing lengthy documents filled with irrelevant information.

3.3 Task Definition

To more effectively achieve the decoupling of knowledge and reasoning, we decompose the training of DRIFT into three distinct stages, each optimized for a different objective, as shown in Figure 2.

3.3.1 Latent Fact Reconstruction Pretraining (LFRP)

We redefine the pretraining objective such that the reasoning model is used only as a frozen decoder to provide a reconstruction signal, while the knowledge model is optimized to generate latent factual representations that best support document reconstruction.

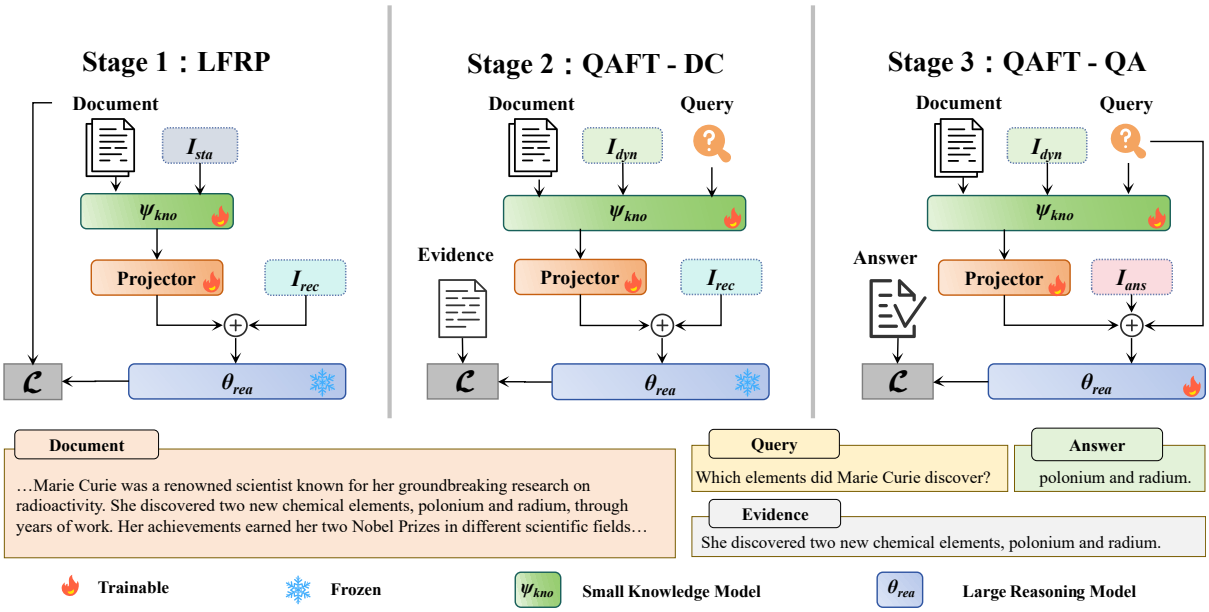

Figure 2: Three different trainging tasks for DRIFT. The instructions in the figure include the dynamic compression instruction, reconstruct instruction, answer instruction, and static compression instructio.

For the knowledge model, this compression is static because it is query-independent. We set the static compression ratio c_{sta} to 8. After applying the bucketed compression strategy, we obtain ξ_{sta}, denoting the number of implicit fact tokens.

Given a document x consisting of n tokens, $X = (x_1, x_2, \ldots, x_n)$, the knowledge model produces latent fact tokens conditioned on a static compression instruction I_{sta}:

$$T_{\text{sta}} = [t_1, t_2, \ldots, t_{\xi_{\text{sta}}}] = \psi_{\text{kno}}(I_{\text{sta}}, x_1, x_2, \ldots, x_N). \tag{5}$$

These implicit fact tokens are projected into fact embeddings via a projector module π:

$$E_{\text{sta}} = \pi(T_{\text{sta}}) = (e_1, e_2, \ldots, e_{\xi_{\text{sta}}}). \tag{6}$$

The reasoning model, parameterized by θ_r, remains frozen during this pretraining stage. It receives a reconstruction instruction I_{rec} and predicts each token conditioned on the fact embeddings and previously generated tokens:

$$\mathcal{L}(\psi_{\text{kno}}, \pi) = -\sum_{x_t \in X} \log P_{\theta_{\text{rea}}}(x_t \mid I_{\text{rec}}, E, x_{<t}), \tag{7}$$

Thus, although the loss is computed using the frozen reasoning model, gradients are only back-propagated through E_{sta} into the projector π and the knowledge model ψ_{kno}. This teaches the knowledge model to produce latent factual representations that are maximally useful for reconstructing the original document.

3.3.2 Query-Aware Fine-Tuning (QAFT) with Single-Context

Through the pretraining objective described above, the knowledge model acquires the ability to encode factual knowledge into a latent space, while the reasoning model learns to understand the implicit fact embeddings. To further adapt the framework for downstream tasks, we introduce an additional training objective that incorporates query inputs. This objective is designed to encourage the knowledge model to implicitly extract and compress query-relevant knowledge into the latent space, and to enable the reasoning model to perform question answering by leveraging the information encoded in the implicit fact embeddings. We train the models on QA datasets, including reading comprehension, fact verification and open-domain question answering.

To better train toward this objective, we perform two fine-tuning tasks in sequence: first a dynamic compression task, followed by a question-answering task.

Dynamic Compression Task In the pretraining task, the knowledge model learns to perform static compression of the context. Now, we aim to train the model to acquire query-aware dynamic compression capability.

In training dataset, each question-answer pair is annotated with supporting evidence. We design a dynamic compression task using the context and

question-evidence pairs: a knowledge model extracts query-specific information from the context into a latent space (implicit fact tokens), while a reasoning model reconstructs the evidence from these tokens. By leveraging the sparsity of query-relevant information within the context, dynamic compression can safely achieve a significantly higher compression ratio than static approaches. The default dynamic compression ratio is set to 32.

We use the instruction I_{dyn} to ask the knowledge model to extract question-relevant knowledge from the document and encodes it into a set of implicit fact tokens in the latent space. Conditioned on the input query Q, the knowledge model ψ_{kno} generates a sequence of question-aware implicit fact tokens T_{dyn}.

The reconstruction performed by the reasoning model under instruction I_{rec} remains analogous to the pre-training stage. However, we diverge by restricting the reconstruction target to the evidence instead of the complete original text. This mechanism employs the evidence as the supervisory signal, explicitly training the knowledge model to develop dynamic compression capabilities.

Given a document $X = (x_1, x_2, \ldots, x_n)$ and a question $Q = (q_1, q_2, \ldots, q_m)$, the knowledge model ψ_k performs dynamic compression on X conditioned on Q.:

$$T_{\text{dyn}} = (t_1, t_2, \ldots, t_{\xi_{\text{dyn}}}) = \psi_{\text{kno}}(I_{\text{dyn}}, X, Q) \quad (8)$$

These latent tokens are then projected into the final fact embeddings $E_{\text{dyn}} = (e_1, e_2, \ldots, e_{\xi_{\text{dyn}}})$ via the projector module π. The reasnoning model still keeps fixed and is asked to reconstruct the evidence X_{evi}:

$$\mathcal{L}(\psi_{\text{kno}}, \pi) = -\sum_{x_k \in X_{\text{evi}}} \log P_{\theta_{\text{rea}}}(x_k \mid I_{\text{rec}}, E_{\text{dyn}}, x_{<k}) \quad (9)$$

Question-answering Task This task is the only one in which the reasoning model is not frozen during training, enabling it to better exploit the compressed context for downstream tasks. We perform fine-tuning by updating the models solely based on the target answers.

The generation of E_{dyn} leverages the same mechanism as the Dynamic Compression Task, implemented by the Knowledge model and projector. Then we instruct the reasoning model with I_{ans} to generate the answer of the question based on the fact embeddings. We denote the answer sequence as $A = (a_1, a_2, \ldots, a_l)$.

We define the training objective as the standard language modeling loss. Our formulation is largely analogous to instruction tuning (Wei et al., 2022), with the key distinction that the context presented to the reasoning model is transformed from the explicit text tokens to the implicit fact embeddings produced by the knowledge model. The question embedding is represented by $E(Q)$. We minimize the following loss to optimize the model parameters:

$$\mathcal{L}(\theta_{\text{rea}}, \psi_{\text{kno}}) = -\sum_{a_j \in A} \log P_{\theta_{\text{rea}}}(a_j \mid I_{\text{ans}}, E_{\text{dyn}}, E(Q), a_{<j}) \quad (10)$$

3.4 Multi-Context Inference without Fine-Tuning

We find that DRIFT, fine-tuned solely in the single-context setting via QAFT, generalizes effectively to multi-context inference without requiring any additional multi-context training. Specifically, to process extensive contexts, we employ an overlapping chunking strategy utilizing `RecursiveCharacterTextSplitter`. We set the chunk size to 8,192 tokens, aligning with the maximum document length used during training. These chunks are then dynamically compressed in parallel by the knowledge model.

3.5 Training Data

We use the English Wikipedia snapshot dated November 1, 2023, treating each entry as a document and its text field as the raw training content. Since longer input documents make model training and convergence more challenging, we adopt a token-level curriculum learning strategy across all three training tasks, with phases defined by input document length. Details of the construction procedures for LFRP and QAFT, as well as the curriculum learning partitions, are provided in Appendix A.

4 Experiments

4.1 Implement Details

For the knowledge–reasoning setup, we fix Qwen2.5-Instruct-3B as the knowledge model and evaluate reasoning models from different families, including Mistral(Jiang et al., 2023) and Qwen2.5(Qwen et al., 2025), with Mistral-7B-Instruct-v0.2 as the default DRIFT backbone. We further explore additional configurations, such as smaller knowledge models (1.5B) and larger reasoning models (14B). All models are trained using

parameter-efficient fine-tuning with LoRA across all tasks, and results for broader model combinations, along with detailed training hyperparameters, are provided in the appendix.

RQ1: To what extent do the latent representations capture and preserve the essential information of the original context?

To assess whether the knowledge model can compress long contexts with minimal information loss, we evaluate the reconstruction fidelity of the learned latent representations before downstream reasoning. Specifically, after training on the LFRP task, we conduct a compression–reconstruction experiment on a test subset with input lengths ranging from 512 to 1024 tokens, where the reasoning model reconstructs the original text conditioned solely on the compressed representations produced by the knowledge model. Reconstruction quality is measured using BLEU and ROUGE (ROUGE-1/2/L) scores.

Models		BLEU	R-1	R-2	R-L
DRIFT after LFRP	Mistral-7B	90.01	94.95	93.93	94.75
	Qwen2.5-7B	83.43	86.60	84.13	88.80
DRIFT before LFRP	Mistral-7B	0.01	1.63	0.07	1.46
	Qwen2.5-7B	0.00	6.32	0.65	4.52

Note: R-1/2/L stands for ROUGE-1/2/L scores. Scores are scaled by 100.

Table 1: Performance evaluation of the reconstruction task across different model configurations.

As illustrated in Table 1, the compression–reconstruction task achieves strong performance across all model combinations following the LFRP training phase. This indicates that after training, the knowledge model effectively compresses the input content and the reasoning model accurately interprets the compressed representations.

RQ 2: How does DRIFT compare to representative baselines in terms of overall effectiveness in long-context reasoning scenarios?

Benchmarks To evaluate DRIFT across diverse long-context scenarios, we employ several representative benchmarks. **BAMBOO** (Dong et al., 2024) serves as a comprehensive suite for testing extended context capabilities through tasks like question answering and code completion. **L-Eval** (Yuan et al., 2025) provides a balanced evaluation of single-hop and multi-hop reasoning across varying document lengths up to 256K tokens while minimizing knowledge leakage. **LongBench-v2** (Bai et al., 2025) consists of challenging multiple-choice questions requiring multi-document understanding and structured data reasoning across contexts reaching millions of words. Finally, **LoCoMo** (Maharana et al., 2024) focuses on long-term conversational memory, probing the model's ability to perform temporal reasoning and event summarization over multi-session dialogue histories.

Metrics For quantitative assessment, we utilize two primary metrics. For datasets involving closed-ended questions or multiple-choice tasks, we employ an **LLM-Judge** to compute accuracy, providing a more robust semantic evaluation than literal matching. For summarization-oriented tasks, we report **ROUGE-L** scores to measure the similarity between generated responses and ground-truth references based on the Longest Common Subsequence.

Baselines We compare **DRIFT** against a diverse set of baseline methods categorized by their compression and retrieval paradigms. For **hard compression**, we select **LLMLingua-2** to represent lexical-level token pruning. For **soft compression**, we evaluate several latent-space models including **ICAE**, **COCOM**, and **xRAG**. Additionally, we implement a **NaiveRAG** baseline utilizing the **BGE-M3** embedding model for document retrieval. The **Mistral-7B-v0.2** model serves as our primary vanilla backbone to provide a performance lower bound without external enhancements.

Results and Analysis Table 2 demonstrates that DRIFT consistently outperforms existing compression-based baselines across diverse long-context benchmarks, particularly under high compression ratios. With the same Mistral backbone, DRIFT sets a new state of the art over existing compression-based methods. Notably, on task-oriented summarization benchmarks (**QMSUM** and **SPACE**) and the **LoCoMo** conversational memory benchmark—where prior compression approaches largely fail—DRIFT remains effective, highlighting its superior ability to preserve and exploit long-range contextual information.

RQ3: How does each training objective contribute to the overall performance of DRIFT?

DRIFT incorporates three distinct training objectives: LFRP, QAFT-DC, and QAFT-QA. To justify the necessity of this multi-stage design and quantify the contribution of each component, we conduct a comprehensive ablation study across three long-context benchmarks.

As shown in Table 3, each training objective in

Model	Comp. Ratio	BAMBOO (16k)				L-Eval (QA Subset)				L-Eval (Sum Subset)			LoCoMo					LongBench-v2			
		AltQA	Meet	Paper	**Avg**	NQ	NarQA	Crse	**Avg**	QMS	SPC	**Avg**	T1	T2	T3	T4	**Avg**	Short	Medium	Long	**Avg**
Baseline Methods based on Mistral-7B-v0.2																					
LLMLingua-2	3×	40.00	75.00	76.77	57.89	77.98	33.18	64.53	58.56	19.37	18.86	19.12	46.81	27.41	48.96	76.93	59.35	26.11	13.95	25.00	20.68
NaiveRAG	–	34.50	79.00	75.76	55.89	72.48	21.50	63.37	52.45	19.61	18.11	18.86	45.04	24.61	41.67	73.48	56.10	17.78	26.05	25.00	22.86
xRAG	128×	25.00	68.00	57.58	43.86	56.88	12.62	50.58	40.03	18.06	12.34	15.20	20.57	6.23	34.38	32.10	24.74	31.11	25.58	25.00	27.43
ICAE	4×	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.67	0.00	0.00	0.59
COCOM	4×	30.50	60.00	47.47	42.11	61.47	9.81	53.49	41.59	9.15	7.79	8.47	9.93	1.87	27.08	10.34	9.55	27.22	23.26	34.26	27.04
	16×	25.50	49.00	41.41	35.34	53.21	8.88	55.81	39.30	7.78	5.37	6.58	11.35	1.87	25.00	10.11	9.55	27.78	23.26	34.26	27.24
	128×	15.00	40.00	36.36	26.57	40.37	5.61	39.53	28.50	8.71	4.79	6.75	12.06	1.56	30.21	8.32	8.96	29.44	21.40	34.26	27.04
DRIFT based on Mistral-7B-v0.2																					
DRIFT (Ours)	**32×**	**41.50**	**80.00**	**77.78**	**60.15**	**69.72**	**27.10**	**61.05**	**52.62**	**21.59**	**19.75**	**20.67**	**36.88**	**23.99**	**33.33**	**80.38**	**57.73**	**32.22**	**28.84**	**25.00**	**29.22**
	64×	**36.00**	**82.00**	**82.83**	**59.15**	**70.64**	**24.77**	**58.72**	**51.38**	**21.95**	**19.49**	**20.72**	**34.75**	**16.82**	**42.71**	**74.67**	**53.31**	**26.67**	**29.77**	**19.44**	**26.44**
	128×	**36.00**	**77.00**	**78.79**	**56.89**	**71.56**	**17.29**	**60.47**	**49.77**	**20.69**	**19.14**	**19.92**	**30.14**	**17.13**	**36.46**	**72.06**	**50.71**	**32.22**	**24.65**	**19.44**	**26.24**
DRIFT based on Qwen2.5-Instruct-7B																					
DRIFT (Ours)	**32×**	**37.00**	**81.00**	**84.85**	**59.90**	**80.73**	**27.10**	**76.16**	**61.33**	**22.66**	**19.01**	**20.84**	**42.55**	**27.73**	**40.62**	**85.49**	**62.79**	**36.67**	**28.37**	**33.33**	**32.41**
Vanilla LLM																					
Mistral-7B-v0.2	1×	40.00	75.00	76.77	57.89	79.82	31.78	65.12	58.91	19.32	18.90	19.11	46.81	27.73	48.96	76.81	59.35	25.00	16.28	23.15	20.87
Qwen2.5-Instruct-7B	1×	36.00	78.00	82.83	58.15	80.73	33.18	75.58	63.16	20.74	18.50	19.62	48.23	38.01	46.88	85.14	66.17	41.11	26.05	24.07	31.01

Table 2: Main results of DRIFT and other baseline methods on long context datasets.

Method	**Bamboo**	**LongBenchv2**	**LoCoMo**
DRIFT (32×)	**59.40**	**28.43**	**57.79**
w/o LFRP	$57.64^{\downarrow 1.76}$	$26.84^{\downarrow 1.59}$	$52.68^{\downarrow 5.11}$
w/o QAFT-DC	$56.64^{\downarrow 2.76}$	$25.84^{\downarrow 2.59}$	$57.36^{\downarrow 0.43}$
w/o QAFT-QA	$45.14^{\downarrow 14.26}$	$18.05^{\downarrow 10.38}$	$36.89^{\downarrow 20.90}$

Note: QAFT-DC denotes the *QAFT Dynamic Compression* objective, and QAFT-QA denotes the *QAFT Question Answering* objective.

Table 3: DRIFT ablation results (avg. accuracy).

DRIFT serves a unique purpose: QAFT-QA provides the fundamental reasoning backbone, while LFRP and QAFT-DC further optimize the latent space efficiency and robustness.

RQ4: Does DRIFT maintain competitive inference efficiency compared to classical baselines?

Efficiency is a critical bottleneck for long-context reasoning. We conduct an end-to-end Time-to-First-Token (TTFT) analysis to evaluate whether DRIFT maintains its performance advantages without incurring prohibitive computational costs as the input scale grows. The end-to-end Token-to-First-Token (TTFT) measures the latency from providing both the input documents and the query to the system until the first output token is generated. This metric directly captures the practical responsiveness of long-context reasoning systems under realistic inference settings.

Figure 3 shows that DRIFT maintains competitive efficiency across all baselines. Notably, the performance gap between DRIFT and the *Full-Context* baseline widens significantly as the input length increases, demonstrating DRIFT's superior scalability for ultra-long sequences.

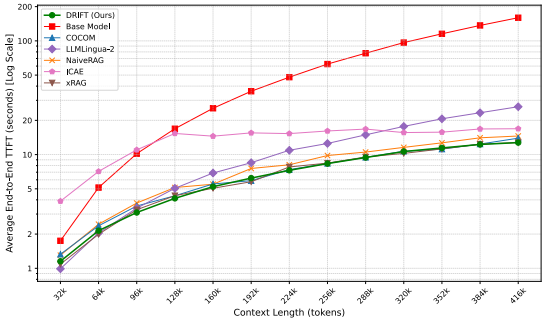

Figure 3: End-to-end TTFT as a function of input length for different baselines.

5 Conclusion

In this paper, we introduced **DRIFT**, a dual-model architecture designed to decouple factual knowledge acquisition from general reasoning in LLMs. Input documents are segmented into chunks, compressed into high-density fact embeddings by a lightweight knowledge model, and interpreted by a larger reasoning model. Experimental results and ablation studies demonstrate the effectiveness of DRIFT and the necessity of each training task. The method also generalizes well across different compression ratios and backbone models, highlighting its robustness and practical applicability.

6 Limitations

Despite its effectiveness, our work has certain limitations that suggest directions for future research. First, due to computational resource constraints, our experiments were primarily validated on models with up to 14B parameters; the performance and scaling laws of DRIFT on larger-scale models remain to be further explored. Second, the use of latent compression introduces challenges regarding interpretability, as the implicit fact tokens are not as human-readable as raw text snippets in traditional RAG. Finally, our current framework is primarily optimized through Supervised Fine-Tuning (SFT). We anticipate that integrating Reinforcement Learning (RL) could further enhance the model's decision-making in knowledge selection and lead to new breakthroughs in performance.

References

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. *Preprint*, arXiv:2412.15204.

Jiaqi Cao, Jiarui Wang, Rubin Wei, Qipeng Guo, Kai Chen, Bowen Zhou, and Zhouhan Lin. 2025. Memory decoder: A pretrained, plug-and-play memory for large language models. *Preprint*, arXiv:2508.09874.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. 2024. xrag: Extreme context compression for retrieval-augmented generation with one token. *Preprint*, arXiv:2405.13792.

LangChain Contributors. 2024. Langchain: A framework for developing applications powered by large language models. GitHub repository. Used the text-splitting utilities (originally from langchain-text-splitters, Version 0.3.8).

Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. *Preprint*, arXiv:2309.13345.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *Preprint*, arXiv:2310.06839.

Yucheng Li, Bo Dong, Chenghua Lin, and Frank Guerin. 2023. Compressing context to enhance inference efficiency of large language models. *Preprint*, arXiv:2310.06201.

Zihan Liao, Jun Wang, Hang Yu, Lingxiao Wei, Jianguo Li, Jun Wang, and Wei Zhang. 2025. E2llm: Encoder elongated large language models for long-context understanding and reasoning. *Preprint*, arXiv:2409.06679.

Fanfan Liu and Haibo Qiu. 2025. Context cascade compression: Exploring the upper limits of text compression. *Preprint*, arXiv:2511.15244.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. Evaluating very long-term conversational memory of llm agents. *Preprint*, arXiv:2402.17753.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

David Rau, Shuai Wang, Hervé Déjean, and Stéphane Clinchant. 2024. Context embeddings for efficient answer generation in rag. *Preprint*, arXiv:2407.09252.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. *Preprint*, arXiv:2109.01652.

Rubin Wei, Jiaqi Cao, Jiarui Wang, Jushi Kai, Qipeng Guo, Bowen Zhou, and Zhouhan Lin. 2025. Mlp memory: A retriever-pretrained memory for large language models. *Preprint*, arXiv:2508.01832.

Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan Qnd Zhuyu Yao, Dahua Lin, Boxun Li, Guohao Dai, Shengen Yan, and Yu Wang. 2025. Lv-eval: A balanced long-context benchmark with 5 length levels up to 256k. *Preprint*, arXiv:2402.05136.

A Data Generation Details

We compute token lengths for Wikipedia documents and sample 100,000 text segments per length bucket, except for the 4k–8k range, where data is insufficient. Each stage was then split into training, validation, and test sets with an 8:1:1 ratio.

A.1 LFRP Data

For the unlabeled data used in pre-training, we deliberately refrain from any data cleaning. Since documents or knowledge bases in real-world scenarios are often noisy and heterogeneous in format, we retain the raw text to improve the robustness of the pre-trained model. For each stage, we use 80,000 samples to train the model.

A.2 QAFT Data

Based on raw Wikipedia documents corresponding to each length bucket, we employed Qwen2.5-72B-Instruct to generate QA pairs, which were subsequently used for fine-tuning in each training stage. In general, each document corresponds to one QA pair; however, for the 4k–8k stage, the available raw documents were insufficient, so some documents were reused. To avoid positional bias in the generated QA data, we segment each document and randomly sample one slice as the input for QA generation. To avoid positional bias in the generated QA data, we segment each document and randomly sample one slice as the input for QA generation.

Data Sampling Strategy To ensure that relevant information is uniformly distributed across the generated dataset, we adopt the following sampling strategy. Each Wikipedia document is first divided into multiple slices of equal length. For every document, we then randomly select one slice as the context for QA generation. This prevents the model from always encountering answers concentrated in specific regions (e.g., the beginning of documents) and ensures that relevant content appears at random positions. As a result, the generated QA pairs cover diverse locations within documents, leading to a more balanced and robust training signal. In addition to generating the question–answer pairs, the model is also required to provide the supporting evidence from the original text corresponding to each answer.

Question Type Diversification To better train the reasoning model to utilize information compressed by the knowledge model, we randomly sample one of three formats—multiple-choice, true/false, or short-answer questions—for QA generation. This exposes the model to diverse reasoning scenarios, thereby strengthening its ability to integrate knowledge-derived information for effective problem solving, and enhancing its generalization across different task settings.

Data Filtering To ensure the quality of the automatically generated QA pairs, we employ Qwen2.5-72B-Instruct as the judger model to filter out low-quality instances. Specifically, the filtering process is guided by five criteria: (i) **Relevance**: the question must be grounded in the given context; (ii) **Correctness**: the provided answer should be factually consistent with the context; (iii) **Clarity**: the question and answer must be well-formed and unambiguous; (iv) **Fidelity**: the evidence must accurately reflect content from the original document without introducing external information; (v) **Sufficiency**: the evidence must provide enough information to answer the question. Only QA pairs that satisfy all conditions are retained for subsequent training.

A.3 Training Strategy: Token-Level Curriculum Learning

During model training, we observed that longer input documents lead to increased training difficulty. In particular, the convergence of both the knowledge and reasoning models becomes more challenging as the number of tokens grows. Therefore, we apply a **token-level curriculum learning strategy** across all three training tasks, where training stages are defined according to the token length of the input documents. Table 4 summarizes the token-level stage configuration for each of the three training tasks.

Task	Stage 1	Stage 2	Stage 3	Stage 4
LFRP	64-128	128-256	256-512	512-1k
QAFT Tasks				
Dynamic Comp.	1k-2k	2k-4k	4k-8k	–
Question Ans.	1k-2k	2k-4k	4k-8k	–

Table 4: Curriculum learning stages. Note that QAFT tasks conclude at Stage 3.

A.4 Construction Prompt

Prompt 1 *"You are a judge evaluating the quality of question-answer pairs. Your task is to determine whether the given answer can be reasonably inferred from the provided evidence.*
Please evaluate based on the following criteria:
1. Can the answer be directly supported by the evidence?
2. Is the evidence sufficient to answer the question?
3. Is the answer logically consistent with the evidence?
4. Are there any contradictions between the answer and evidence?
Question: {question}
Evidence: {evidence}
Answer: {answer}
Please respond with only "true" if the answer can be reasonably inferred from the evidence, or "false" if it cannot.
Your judgment:",
Prompt 2 *" Please generate a question that can be answered based on the provided context. The question should be highly relevant to the context, and the answer must be directly inferable from the given information. Avoid asking questions that cannot be answered using the context. The question should be of the type: {question_type}.*
Your response should consist of three parts:
1. Question – the generated question. (a string)
2. Answer – the answer, including how it is reasoned out from the relevant information in the context. (a string)
3. Evidence – the specific part(s) of the original text that support the answer. (a string)
Attention: Evidence must be quoted directly from the original text and must include all the information needed to answer the question. If some parts of the evidence involve unclear references (e.g., ambiguous subjects), include the related sentences that clarify them, so that the evidence alone is sufficient for answering the question. Ensure that every sentence remains complete, without the use of ellipses.
Your output format should be:
json
{{
"question": "<the generated question (include options if the question type is multiple choice)>",
"answer": "<the corresponding answer, including how it is inferred from the relevant information in the context>",
"evidence": "<the specific part(s) taken directly from the original text that support the answer>"
}}
Context: {context}
Your output:
",

B Dataset Statistics

Dataset Statistics and Configuration: Table 5 summarizes the statistics of our cleaned raw dataset. The dataset comprises approximately 2.13 million samples with a total of 1.61 billion tokens, covering a wide range of sequence lengths from 64 to 4,096. This diverse distribution ensures the model's robustness across various context windows. For the training pipeline, we strategically allocated the data across different phases: the LFRP stage utilized 640,000 samples to establish solid feature representations, while the QAFT stage employed 240,000 specifically constructed samples to refine the model's task-specific performance.

Table 5: Statistics of the dataset across different token length ranges.

Range	Train Samples	Val / Test Samples	Total Samples	Total Tokens
64 – 128	320,000	40k / 40k	400,000	38,138,338
128 – 256	320,000	40k / 40k	400,000	76,466,097
256 – 512	320,000	40k / 40k	400,000	153,042,482
512 – 1,024	320,000	40k / 40k	400,000	305,885,575
1,024 – 2,048	303,724	37k / 37k	379,656	580,138,267
2,048 – 4,096	118,272	14k / 14k	147,840	451,470,489
Grand Total	**1,701,996**	**212k / 212k**	**2,127,496**	**1,605,141,248**

C Training Details

C.1 Hyperparameter Settings

The specific hyperparameter configurations for our model architectures and training environment are summarized here. We conduct the training using Low-Rank Adaptation (LoRA) to ensure parameter efficiency. Specifically, we set the LoRA rank to r=16 and the scaling factor to =32, incorporating a dropout rate of 0.05 to mitigate overfitting. The optimization is performed with a learning rate of 0.0001 and a total effective batch size of 128.

C.2 Training Loss Curves Across Three Stages

To evaluate the optimization stability and convergence of DRIFT, we illustrate the training loss trajectories across its three sequential stages below.

C.2.1 Stage 1: Latent Fact Reconstruction Pretraining (LFRP)

The training loss for the LFRP stage is presented in Figure 4. The curve shows a steady decline and eventual plateau, indicating that the knowledge model successfully learned to reconstruct factual content into the latent space with high fidelity.

C.2.2 Stage 2: Query-Aware Fine-Tuning (QAFT) with Single-Context Dynamic Compression

During this stage, we fine-tune the model on the Dynamic Compression Task. The training objective focuses on the model's ability to compress and project query-relevant knowledge into the reasoning model's embedding space. The loss reflects

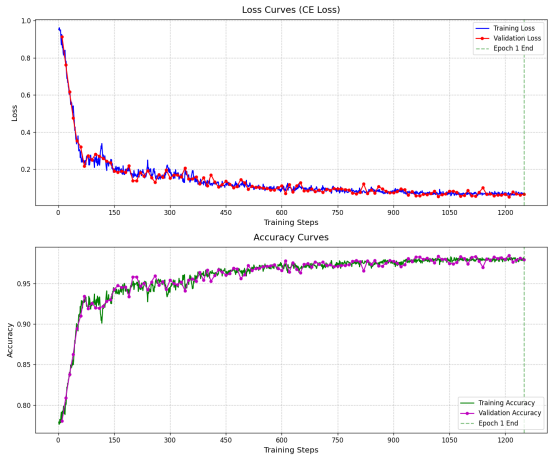

Figure 4: Training loss trajectory of Stage 1

the efficiency of information bottlenecking under query guidance.

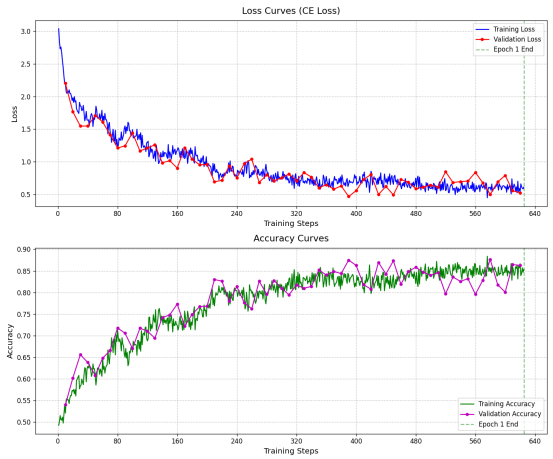

Figure 5: Training loss trajectory of Stage 2

C.2.3 Stage 3: Query-Aware Fine-Tuning (QAFT) with Single-Context Question Answering

The final stage involves end-to-end optimization for the Question-Answering (QA) task. We report the cross-entropy loss during this phase, which demonstrates how the reasoning model effectively utilizes the distilled latent facts to generate accurate and context-grounded answers.

D Additional Experimental Results

D.1 General-Purpose Capability Comparison

To assess whether the reasoning-oriented fine-tuning affects the model's broad utility, we compare the general-purpose capabilities of the reasoning model before and after our proposed training. The results indicate that the degradation in generic competencies remains minimal. This suggests that our

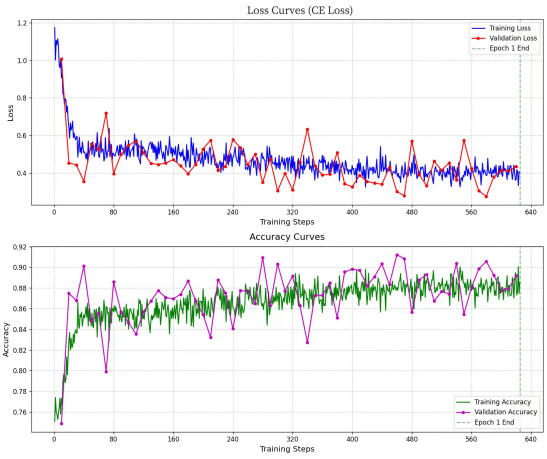

Figure 6: Training loss trajectory of Stage 3

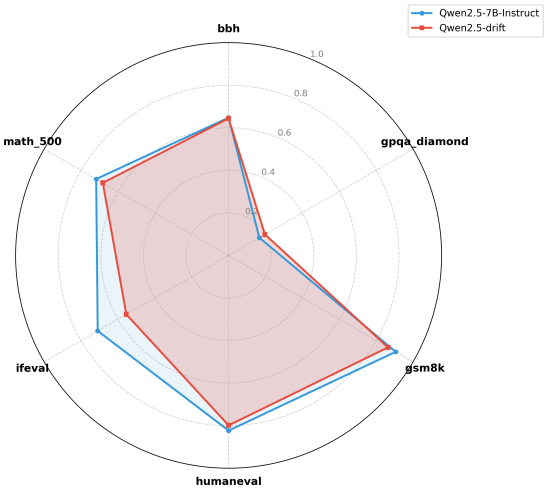

Figure 7: Placeholder radar chart illustrating general-purpose capabilities before and after reasoning-oriented fine-tuning.

fine-tuning strategy effectively enhances reasoning performance while preserving overall versatility.

D.2 Effects of Model Size Combinations

Experimental analysis of the performance impact when using different size combinations for the knowledge and reasoning models.

Table 6: Performance of different model combinations on LongBenchv2.

Combination	LongBenchv2
Qwen-7B × 3B	32.41
Qwen-7B × 1.5B	31.81
Qwen-14B × 3B	34.39

RQ3: How does the model align its latent reasoning with explicit evidence during training?

A critical question in our framework is whether the reasoning model merely replicates explicit textual evidence or instead develops distinct and effi-

cient reasoning strategies within the latent space. This distinction is crucial for determining whether the implicit context functions as a complementary modality rather than a redundant compression.

To examine this behavior, we introduce a diagnostic metric, **Reasoning Consistency** (M_{ED}), which is monitored during the question-answering task in the **QAFT stage**. Specifically, M_{ED} measures the Kullback–Leibler (KL) divergence between the output distributions of the reasoning model when conditioned on compressed latent representations versus explicit textual evidence:

$$M_{\text{ED}} = D_{\text{KL}}\Big(P_{\theta_{\text{rea}}}(\cdot \mid I_{\text{ans}}, E_{\text{dyn}}, \text{E}(Q)) \,\Big\|\, P_{\theta_{\text{rea}}}(\cdot \mid I_{\text{ans}}, X_{\text{evi}}, Q)\Big) \tag{11}$$

Importantly, M_{ED} is used solely as a non-intrusive diagnostic probe to analyze reasoning behavior and does not participate in gradient backpropagation.

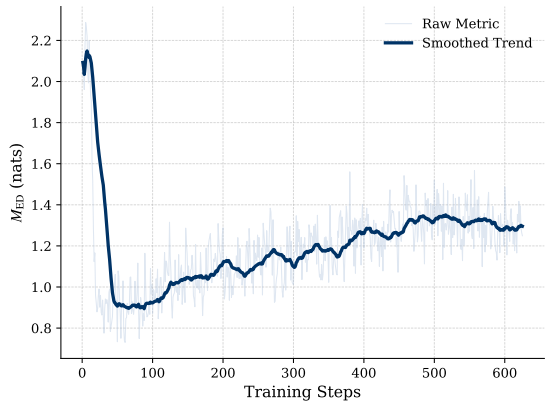

Figure 8: Evolution of the reasoning consistency metric M_{ED} during traiwoqning.

The trajectory of M_{ED} in Figure 8 exhibits a clear *Grounding-then-Specializing* learning pattern with two phases. In **Stage I (Rapid Grounding)**, M_{ED} decreases sharply, indicating that the reasoning model initially aligns its latent representations closely with explicit textual evidence to ensure faithful reasoning. In **Stage II (Emergent Specialization)**, M_{ED} gradually increases and stabilizes, suggesting that the model moves beyond strict textual alignment and develops more specialized and efficient reasoning strategies in the latent space. Overall, this behavior shows that the implicit context is not a mere compressed copy of the input, but a complementary modality that supports task-oriented reasoning beyond surface-level text.