

CheckRLM: Effective Knowledge-Thought Coherence Checking in Retrieval-Augmented Reasoning

Dingling Xu^{12*}, Ruobing Wang^{34*}, Qingfei Zhao³⁴, Yukun Yan^{5†}, Zhichun Wang^{12†}, Daren Zha³, Shi Yu⁵, Zhenghao Liu⁶, Shuo Wang⁵, Xu Han⁵, Maosong Sun⁵

¹School of Artificial Intelligence, Beijing Normal University, Beijing 100875, China

²Beijing Key Laboratory of Artificial Intelligence for Education, Beijing 100875, China

³Institute of Information Engineering, Chinese Academy of Sciences,

⁴School of Cyber Security, University of Chinese Academy of Sciences,

⁵Department of Computer Science and Technology, Institute for AI, Tsinghua University,

⁶Northeastern University

Abstract

Reasoning Language Models (RLMs) have significantly improved performance on complex tasks by extending the reasoning chain. However, these chains are prone to containing factual errors, particularly in knowledge-intensive tasks. To address this issue, we propose **CheckRLM**, a framework that improves the reliability of the reasoning process through Retrieval-Augmented Generation (RAG) by timely checking and correcting factual errors. Specifically, CheckRLM extracts factual claims from the reasoning chain to identify and localize subtle knowledge inconsistencies during inference. Upon detection of errors, a refinement mechanism performs minimal-cost yet precise corrections by leveraging external knowledge, ensuring coherence between the reasoning chain and correct knowledge. Extensive experiments demonstrate that CheckRLM substantially outperforms existing baselines, exhibiting a strong capability to mitigate error accumulation in long-horizon reasoning with lower costs. The code and data are available at <https://github.com/AI9Stars/CheckRLM>.

1 Introduction

Reasoning Language Models (RLMs) such as OpenAI-o1 (OpenAI, 2024) and DeepSeek-R1 (Guo et al., 2025) have notable advanced reasoning capabilities with reinforcement learning and test-time scaling (Zhang et al., 2026; Gao et al., 2025). By decomposing complex problems and engaging in iterative exploration and reflection over long reasoning chains, RLMs attract significant attention in logic-intensive domains, including mathematics and coding (Chen et al., 2025; Li, 2025).

Despite these advancements, the extended reasoning chain also introduces the critical challenge of maintaining factual coherence throughout the

*Equal contribution. 202111081066@mail.bnu.edu.cn

†Corresponding authors. yanyk.thu@gmail.com; zcwang@bnu.edu.cn

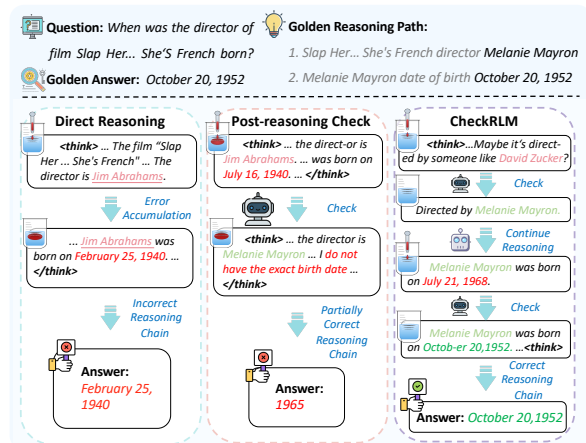


Figure 1: **Illustration of error accumulation and CheckRLM.** Direct Reasoning and Post-reasoning Check suffer from erroneous internal knowledge in the RLM and insufficient external knowledge, whereas CheckRLM corrects errors timely to reach the correct answer.

process, especially in knowledge-intensive tasks (He et al., 2025; Yao et al., 2025). In long step-wise reasoning, we observe that the factual accuracy of each intermediate step is paramount, as it forms the foundational premise for subsequent deductions. Consequently, any factual error at a given step can be propagated, leading to final answer deviations—a phenomenon termed error accumulation (Ling et al., 2023; Tyen et al., 2024). As illustrated in Figure 1, Direct Reasoning initially produces an incorrect director name “Jim Abrahams”, which is then propagated as a premise for subsequent reasoning, leading to an erroneous inference of the director’s birth date and the final incorrect answer “February 25, 1940”. Although Post-reasoning Check later revises the director’s name, it fails to repair the already corrupted intermediate reasoning.

To bridge this gap, we propose **CheckRLM**, a framework that employs Retrieval-Augmented Generation (RAG) to promptly identify and cor-

rect factual errors within long reasoning chains, thereby aligning them with external knowledge. To effectively check and refine reasoning chains, CheckRLM comprises two components: in-process knowledge claim recognition and localized knowledge coherence correction via retrieval. During in-process knowledge claim recognition, key factual claims pertinent to the query are identified and extracted from each newly generated reasoning segment, facilitating the location of potential inconsistencies while minimizing interference from irrelevant information. In the localized knowledge coherence correction via retrieval process, we use the extracted key factual claims along with the original question to search an external knowledge base and perform token-level precise corrections, ensuring that each segment of the verified reasoning chain aligns with external knowledge and reducing error accumulation.

In summary, our contributions are as follows: 1) we propose CheckRLM, an efficient RAG framework with in-reasoning knowledge checking that identifies and corrects factual errors during long-chain reasoning, effectively preventing error accumulation; 2) we introduce a localized intermediate intervention mechanism that dynamically extracts and verifies critical knowledge claims from partial reasoning states, enabling timely and fine-grained factual correction; 3) CheckRLM significantly outperforms recent strong baselines, while comprehensive analyses show that it enables early, targeted error correction and simultaneously achieves top-tier performance with significantly reduced inference time and token consumption.

2 Related Work

2.1 Reasoning Language Model (RLM)

RLMs enhance the quality of final answers by guiding Large Language Models (LLMs) to decompose complex tasks into multiple subquestions and by extending step-by-step reasoning chains (Chen et al., 2025; Gao et al., 2025; Zhang et al., 2026; Li, 2025). Prior work primarily relies on engineered prompts during inference without modifying the internal parameters of LLMs (Yasunaga et al., 2024; Zhou et al., 2023; Zhang et al., 2023; Yao et al., 2023). Chain-of-Thought (Wei et al., 2022) inserts several task-oriented exemplars that contain intermediate reasoning steps into the prompt, while (Kojima et al., 2022) stimulates zero-shot generation of intermediate reasoning by adding the in-

struction “*let’s think step by step*” to task instructions, effectively eliciting reasoning capabilities in LLMs. However, these prompt-engineering approaches are largely constrained by the quality of the prompts and exemplars, and the depth and correctness of the elicited reasoning are difficult to guarantee. Recent studies (OpenAI, 2024; Guo et al., 2025; Yang et al., 2025) have focused on deep optimization of LLMs via large-scale reinforcement learning (RL). By designing verifiable reward functions, RLMs exhibit high-level reasoning behaviors, including self-verification, reflection, and dynamic strategy adjustment; furthermore, increasing test-time compute yields continued improvements in reasoning performance (Weng et al., 2023; Kumar et al., 2025). Meanwhile, concurrent studies have pointed out that these RLMs tend to produce hallucinations within their long reasoning chains (He et al., 2025; Yao et al., 2025). An early factual error can subsequently mislead the entire reasoning trajectory, leading to a phenomenon known as “error accumulation” (Ling et al., 2023; Tyen et al., 2024).

2.2 Retrieval-Augmented Generation (RAG)

RAG effectively mitigates the “hallucination” in knowledge-intensive tasks by integrating LLMs with external knowledge bases, improving the accuracy and reliability of content (Lewis et al., 2020; Ram et al., 2023; Zhao et al., 2024; Zhu et al., 2025). However, the single-retrieval mechanism in vanilla RAG (Izacard et al., 2023) often yields limited information, struggling to cover the complete knowledge background required for complex questions (Wang et al., 2025, 2024). To address the limitations of Vanilla RAG, researchers have proposed the adaptive RAG framework (Jeong et al., 2024; Jiang et al., 2023; Asai et al., 2024; Yao et al., 2022), which employs an intelligent decision-making mechanism to dynamically determine when to retrieve, what to retrieve, and how to integrate the retrieved information. This significantly improves both the response quality and efficiency of RAG systems. However, these traditional adaptive RAG methods are not suitable for RLMs based on long chains of thought. Concurrent work (Hu et al., 2025; Sun et al., 2025; Guan et al., 2026; Xiong et al., 2025; Zhao et al., 2025; Li et al., 2025) combines reasoning with search, inserting external knowledge into long reasoning chains to augment RLMs’ reasoning ability. These methods integrate external knowledge

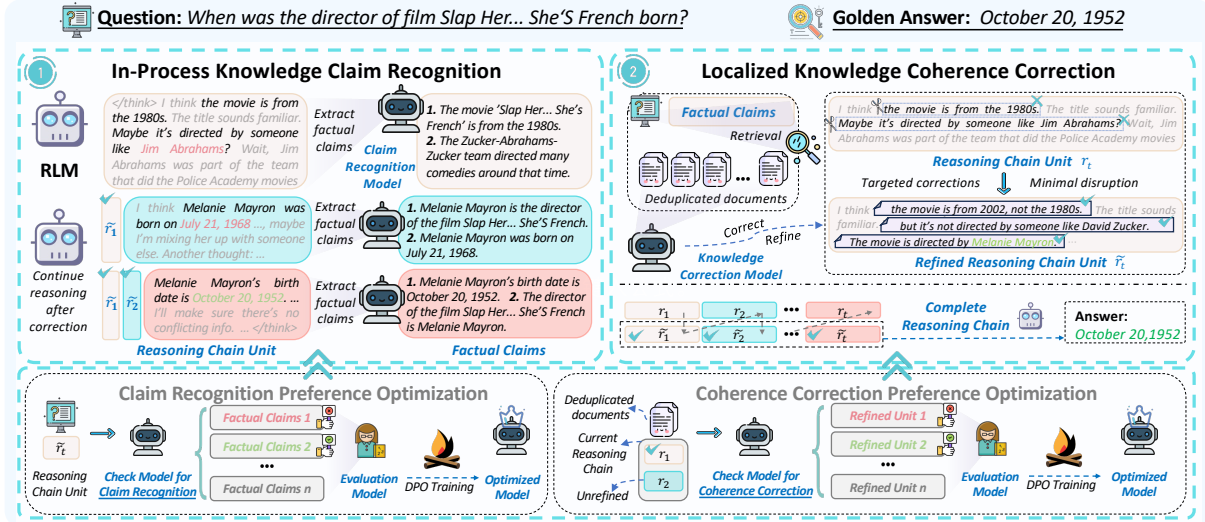


Figure 2: **Overview of CheckRLM.** In-process knowledge claim recognition locates error positions by extracting factual claims from reasoning chain units, while localized knowledge coherence correction corrects factual errors at minimal cost based on retrieved documents. The two modules are jointly optimized using DPO.

in a coarse-grained manner, which can introduce instability in the coherence of the reasoning process. Instead, we propose a minimal intervention approach that corrects erroneous knowledge within the model itself, achieving more reliable and controllable knowledge-augmented reasoning capabilities.

3 Methodology

3.1 Preliminary of CheckRLM

Considering a complicated question q that requires multi-step reasoning, our goal is to generate a correct final answer a by checking and refining the reasoning chain \mathcal{R} generated by RLMs. Mathematically, the reasoning chain \mathcal{R} can be defined as a sequence $\tau = (s_1, s_2, \dots, s_T)$, where each s_t represents an intermediate reasoning state generated at time step t . The model parameters θ govern the conditional distribution $P_\theta(s_t | q, s_{<t})$, denoting the probability of generating the subsequent reasoning step s_t conditioned on the question q and the preceding reasoning history $s_{<t}$. Consequently, the sampling process for the entire reasoning chain \mathcal{R} can be formalized as the product of stepwise conditional distributions:

$$\mathcal{R} \sim \prod_{t=1}^T P_\theta(s_t | q, s_{<t}) \quad (1)$$

The final answer a is then derived from both \mathcal{R} and q , where $P_\theta(a | q, \mathcal{R})$ represents the conditional distribution under which the reasoning model

yields the final answer:

$$a \sim P_\theta(a | q, \mathcal{R}) \quad (2)$$

3.2 Knowledge–Thought Coherence Checking

In this section, we first introduce a two-stage Knowledge–Thought Coherence Checking process (§3.2.1 and §3.2.2), and then describe the training data construction pipeline and the training process (§3.2.3). The overall framework of CheckRLM is shown in Figure 2.

3.2.1 In-Process Knowledge Claim Recognition

We adopt an “*intermediate intervention*” strategy to extract explicit knowledge claims during the reasoning process, rather than intervening only after the complete reasoning chain is generated. This strategy aims to address potential factual errors and prevent the gradual amplification of randomly occurring erroneous knowledge in the intermediate states of the reasoning chain at an early stage.

Specifically, we identify and extract factual information that supports problem-solving from the reasoning chain, summarizing it into multiple explicit factual claims. These factual claims serve the fact verification model \mathcal{M}_{rec} to prioritize question-relevant facts and facilitate the detection of explicit errors for correction, preventing their propagation and negative impact on downstream reasoning. Considering that granular checks, such as token-level or sentence-level, disrupt the logic and semantic meaning of the reasoning chain and

trigger unnecessary or frequent interruptions, we perform coherence checks at the paragraph level, keeping a balance between contextual completeness and timely verification.

Accordingly, the current reasoning chain unit can be defined as the chain generated from s_{t-1} to s_t , where the time step t is segmented by paragraphs in the reasoning chain and denoted as r_t . We input the original question q and the current reasoning chain unit r_t into the claim recognition model \mathcal{M}_{rec} to sample a set of knowledge claims y_t^{claim} . For robustness considerations, we do not use the entire previously generated reasoning chain $\mathcal{R}_{<t}$ as input during recognition to avoid introducing noise from earlier reasoning steps. We formalize In-Process Knowledge Claim Recognition as follows:

$$y_t^{\text{claim}} \sim \mathcal{M}_{\text{rec}}(\cdot \mid \text{Instruct}_r, r_t, q) \quad (3)$$

In this formulation, $y^{\text{claim}} = \{y_{t_1}, \dots, y_{t_n}\}$ represents the set of knowledge claims generated by the claim recognition model \mathcal{M}_{rec} , where $n \in \mathbb{N}^+$ denotes the number of generated claims, and Instruct_r is a prompt template designed to guide the model to focus on factual claims relevant to the q .

3.2.2 Localized Knowledge Coherence Correction via Retrieval

At the In-Process Knowledge Claim Recognition process, we extract a set of factual claims y_t^{claim} related to the current reasoning chain unit r_t , which helps identify specific locations where factual knowledge errors may exist. Subsequently, the Localized Knowledge Coherence Correction via Retrieval process focuses on correcting the reasoning chain unit r_t at these identified locations by retrieving external knowledge, effectively aligning internal reasoning chain content with external knowledge and correcting parametric knowledge errors.

Specifically, we first take the union of the knowledge claim set y_t^{claim} and the original question q to construct the query set $\mathcal{Q}_t = \{q\} \cup y_t^{\text{claim}}$. Each query $q_i \in \mathcal{Q}_t$ serves as an atomic retrieval query, triggering an independent retrieval that returns the top- k relevant documents. Collecting the results from all queries forms the initial candidate document set $\mathcal{D}_t^{\text{raw}}$. To avoid redundancy, we further perform deduplication on the initial candidate document set $\mathcal{D}_t^{\text{raw}}$ to obtain the refined retrieval document set $\mathcal{D}_t = \{d_{t_1}, d_{t_2}, \dots, d_{t_n}\}$, where $n \in \mathbb{N}^+$

denotes the number of documents after deduplication. The process for acquiring external knowledge can be represented in a formal notation as follows:

$$\mathcal{D}_t = \bigcup_{q_i \in \mathcal{Q}_t} \text{Retriever}(q_i) \quad (4)$$

Then, we jointly feed the retrieved document set \mathcal{D}_t and the current reasoning-chain unit r_t into the knowledge correction model \mathcal{M}_{cor} . The model identifies and extracts knowledge relevant to r_t and uses it to assess and verify the factual validity of the current reasoning-chain unit r_t , ensuring knowledge alignment between the reasoning chain and the knowledge base. Conditioned on the current reasoning-chain unit r_t and the retrieved document set \mathcal{D}_t after deduplication, the corrected reasoning-chain unit r'_t is sampled from the conditional distribution parameterized by the knowledge correction model \mathcal{M}_{cor} :

$$r'_t \sim \mathcal{M}_{\text{cor}}(\cdot \mid \text{Instruct}_c, r_t, \mathcal{D}_t) \quad (5)$$

where Instruct_c is an instructional prompt template designed to guide the model in performing token-level corrections with minimal cost while ensuring the reasoning structure remains intact. Here, we introduce two types of preference guidance to steer the model, such that when all facts in r_t are correct or the retrieved document set \mathcal{D}_t is irrelevant, the reasoning-chain unit remains unchanged. If r_t contains erroneous knowledge, the model applies targeted token-level corrections with minimal disruption to preserve the structural integrity of the reasoning chain, ultimately producing the corrected reasoning-chain unit r'_t . We define $\tilde{r}_t \in \{r_t, r'_t\}$ to uniformly represent the current reasoning chain that admits two possible content states.

After factual errors are corrected, the reasoning model continues the inference process based on the updated reasoning chain, enabling the model to proceed along a correct reasoning path and preventing the accumulation of errors. Through multiple iterations of reasoning and factual correction, the complete reasoning chain is ultimately denoted as:

$$\mathcal{R} = \tilde{r}_1 \oplus \tilde{r}_2 \oplus \dots \oplus \tilde{r}_t \quad (6)$$

from which the final answer a is derived. During reasoning, this real-time factual verification and correction mechanism leverages external knowledge to precisely identify and rectify parametric knowledge errors in the RLM, guiding the model toward accurate and efficient reasoning.

3.2.3 Optimization for Knowledge Recognition and Correction

Although the two processes above can effectively identify and correct factual errors during reasoning, we also observe that the claim recognition model \mathcal{M}_{rec} and the knowledge correction model \mathcal{M}_{cor} suffer from issues in a limited number of cases, such as low-quality factual claims, failure to perform correct corrections, and redundant outputs. Therefore, our goals are twofold: 1) to improve the factual claim recognition capability of the \mathcal{M}_{rec} , and 2) to enhance the precision of the \mathcal{M}_{cor} in performing knowledge corrections. To this end, we adopt DPO (Rafailov et al., 2023) as the optimization algorithm to guide the model in assigning higher likelihood to high-quality, accurate, and concise outputs, without requiring an explicit reward model. Accordingly, we adopt a two-stage strategy to construct a small-scale, high-quality preference dataset \mathcal{D}_{DPO} (Appendix A), with the aim of inducing preferences for high-quality factual claim recognition and fine-grained knowledge correction. **DPO Training** Using the constructed training data, we optimize the model $\mathcal{M}_{\text{RC}}^\theta$ that serves both the claim recognition and knowledge correction, to learn a unified policy θ that assigns higher likelihood to the preferred output y^+ than to its less preferred counterpart y^- , conditioned on the same input x . Our training objective is formulated as follows:

$$\mathcal{L}_{\text{DPO}}(\mathcal{M}_{\text{RC}}^\theta; \mathcal{M}_{\text{RC}}^{\text{ref}}) = -\mathbb{E}_{\{x, y^+, y^-\} \sim \mathcal{D}} [\log \sigma \left[\beta \log \frac{\mathcal{M}_{\text{RC}}^\theta(y^+ | x)}{\mathcal{M}_{\text{RC}}^{\text{ref}}(y^+ | x)} - \beta \log \frac{\mathcal{M}_{\text{RC}}^\theta(y^- | x)}{\mathcal{M}_{\text{RC}}^{\text{ref}}(y^- | x)} \right]] \quad (7)$$

where $\mathcal{M}_{\text{RC}}^\theta$ and $\mathcal{M}_{\text{RC}}^{\text{ref}}$ represent the trainable model and the reference model, respectively.

4 Experiments

4.1 Datasets & Metrics

Multi-hop QA Datasets (1) **HotpotQA** (Yang et al., 2018) requires retrieving and reasoning over multiple Wikipedia documents with distracting information to answer. (2) **2WikiMultiHopQA (2WikiMQA)** (Ho et al., 2020) is a multi-hop QA dataset where each question requires reasoning over entity-centric hyperlinks across multiple Wikipedia pages to discover bridging entities. (3) **MuSiQue** (Trivedi et al., 2022) is a challenging dataset characterized by its compositional, interconnected questions that necessitate sequential reason-

ing without relying on isolated single-hop clues. (4) **IIRC** (Ferguson et al., 2020) builds on Wikipedia articles where questions require identifying relevant text snippets from linked contexts to answer questions given an incomplete context. For the test samples and retrieval corpus, we follow IRCot (Trivedi et al., 2023).

Short-form QA Dataset focuses on retrieving direct answers to factoid questions from a single context. We choose **SimpleQA** (Wei et al., 2024), a large-scale benchmark consisting of short, fact-seeking questions. We randomly select 500 samples from the test split and use the KILT (Petroni et al., 2021) knowledge source as the corpus.

Metrics We use F1-Score (f1) and Exact Match (em) (Jiang et al., 2023) to evaluate the correctness of the final answer.

4.2 Baselines

We compare our method against the following baseline approaches: (1) **Direct Reasoning**: the RLM relies solely on its parametric knowledge to answer questions without retrieving any external information. (2) **Vanilla RAG**: a single-step retrieval fetches the top- k most relevant documents, which are then provided alongside the question as context for the LLM to generate an answer. (3) **Multi-Step RAG**: the RAG system gathers relevant information through multiple iterative retrieval steps. We select RAT (Li et al., 2024) as our representative baseline. (4) **Adaptive RAG**: the RAG framework dynamically decides the retrieval and generation strategy to optimize both response quality and efficiency. We primarily compare against three established baselines: FLARE (Jiang et al., 2023), Self-RAG (Asai et al., 2024), and ReAct (Yao et al., 2022). Additionally, we also compare with Search-o1 (Li et al., 2025), which actively retrieves external knowledge during the reasoning process and seamlessly integrates refined, concise information extracted from lengthy retrieval results into the reasoning chain.

4.3 Implementation Details

We primarily employ QwQ-32B (Team, 2025) as the reasoning model, while also evaluating various RLMs of Qwen3-8B (Yang et al., 2025), Qwen3-32B (Yang et al., 2025) and DeepSeek-R1-Distill-Llama-70B (Guo et al., 2025) for comparative analysis. For \mathcal{M}_{rec} and \mathcal{M}_{cor} , we use the same underlying model and select four LLMs: Qwen3-8B (Yang et al., 2025), Qwen2.5-14B-Instruct (Yang

Method	Multi-hop								Short-form		Avg.
	HotpotQA		2WikiMQA		MuSiQue		IIRC		SimpleQA		
	f1	em	f1	em	f1	em	f1	em	f1	em	
<i>QwQ-32B</i>											
Direct Reasoning	38.4	29.0	34.6	28.4	18.5	7.4	24.8	21.4	10.5	4.6	21.8
Vanilla RAG	52.7	42.2	46.4	42.6	19.3	10.0	25.0	22.0	31.4	24.2	31.6
ReAct (2022)	48.2	34.6	45.9	33.0	22.3	10.0	21.9	16.2	30.1	22.2	28.4
FLARE (2023)	43.5	32.2	46.6	38.4	18.4	9.2	12.3	9.6	24.1	17.4	25.2
Self-RAG (2024)	49.7	39.9	40.0	36.2	20.0	10.6	25.1	21.3	31.1	24.8	29.9
RAT (2024)	51.5	39.0	47.2	38.6	24.1	13.2	21.7	17.0	30.3	22.2	30.5
Search-o1 (2025)	62.0	49.2	71.4	60.4	33.3	20.7	29.2	25.0	35.4	27.4	41.5
CheckRLM	66.3	52.6	73.4	62.0	39.6	27.2	33.1	29.0	40.0	30.4	45.4
<i>Qwen3-32B</i>											
Direct Reasoning	36.4	27.6	34.0	29.4	16.6	6.6	23.0	20.2	9.5	2.4	20.6
Vanilla RAG	51.2	41.0	44.5	41.8	19.7	10.4	22.5	19.8	29.3	22.8	30.3
CheckRLM	64.2	52.0	70.8	60.8	35.6	25.4	31.6	27.8	39.1	30.6	43.8
<i>Qwen3-8B</i>											
Direct Reasoning	29.3	22.2	31.1	26.6	12.9	5.2	21.0	18.2	7.7	2.8	17.7
Vanilla RAG	44.5	35.2	40.9	38.4	13.8	6.0	19.8	16.8	29.4	23.0	26.8
CheckRLM	61.9	49.6	69.7	58.6	35.0	24.0	29.7	25.4	35.7	28.6	41.8

Table 1: **Results (%) of overall performance.** Our recognition and correction model is Llama-3.3-70B-Instruct. The highest scores are emphasized in **bold**.

et al., 2024), Qwen2.5-32B-Instruct (Yang et al., 2024), and Llama-3.3-70B-Instruct (Dubey et al., 2024). In terms of inference hyperparameters, we set the maximum context length to 16,384 tokens and retain default settings for all other hyperparameters. For the retriever, we utilize both the sparse retriever BM25 and the dense retriever bge-large-en-v1.5 (Xiao et al., 2023) on multi-hop QA datasets, while only the dense retriever bge-large-en-v1.5 is used on the short-form QA dataset. Across all experiments, we fix top- k to 3 and set the maximum number of recognition-and-correction steps to 10. All experiments are conducted using the vLLM (Kwon et al., 2023) on a cluster of 8xA800 GPUs. For DPO training, we use a batch size of 8, a learning rate of $5e-7$, β of 0.1, and train for 1 epoch.

5 Results and Analysis

5.1 Overall Performance

Vanilla RAG underperforms on complex tasks; by contrast, CheckRLM demonstrates a substantial performance improvement. As shown in Table 1, Vanilla RAG improves upon Direct Reasoning

by introducing external knowledge documents in a one-time retrieval step, thus partially alleviating the limitation that Direct Reasoning relies entirely on internal knowledge during the reasoning process. Empirical results indicate that Vanilla RAG achieves reasonable performance on short-form QA tasks, as these tasks typically require only a small number of relevant snippets that are most relevant to the original question; however, these observed performance improvements struggle to generalize to complex multi-hop QA tasks. In contrast, CheckRLM consistently outperforms Vanilla RAG across all datasets. In particular, it achieves especially substantial gains on complex multi-hop QA tasks, yielding improvements over Vanilla RAG across all three reasoning backbones. These performance improvements indicate that CheckRLM is capable of performing effective knowledge correction by repeatedly and timely incorporating external knowledge to precisely correct factual errors in the reasoning chain.

CheckRLM surpasses a wide range of mainstream baselines across both multi-hop and short-form QA benchmarks. As shown in Table 1, in comparison with recent iterative re-

Method	Hot.	2Wiki.	Simp.
<i>QwQ-32B</i>			
Vanilla RAG	52.7	46.4	31.4
Inference Only	61.6	65.5	36.0
DPO Training	63.2	71.2	36.9
<i>Qwen3-32B</i>			
Vanilla RAG	51.2	44.5	29.3
Inference Only	57.0	62.9	34.2
DPO Training	61.3	65.3	36.1

Table 2: **DPO Training Results (%)**. We conduct experiments on the HotpotQA, 2WikiMQA and SimpleQA dataset, using Qwen2.5-14B-Instruct as the recognition and correction model and f1 as the metric.

trieval-generation and coupled retrieval-reasoning methods, CheckRLM achieves the highest scores across all evaluated datasets, including challenging multi-hop and short-form QA benchmarks. The gains are especially pronounced on complex multi-hop reasoning, and are most evident on the more challenging MuSiQue dataset, where CheckRLM outperforms the best-performing baseline by 6.3 f1 points. This advantage can be attributed to a key design difference. While these prior methods enhance reasoning by expanding the interaction between retrieval and reasoning or by increasing search depth, they generally lack explicit mechanisms to correct factual inconsistencies within the reasoning process itself. In contrast, CheckRLM introduces an in-process knowledge checking mechanism that continuously monitors and corrects factual errors during reasoning, preventing erroneous knowledge from propagating across subsequent reasoning steps. This design is especially critical for multi-hop reasoning, where early-stage errors can severely affect downstream inference.

5.2 Effectiveness of Knowledge Recognition and Correction Optimization

The DPO training process in CheckRLM enhances the model’s ability to correct factual errors in the reasoning chain by jointly optimizing knowledge claim recognition and knowledge coherence correction. Based on this design, we evaluate the performance gains brought by this joint optimization strategy to validate the effectiveness of our training procedure. We conduct DPO training on the small backbone Qwen2.5-14B-Instruct, and the results are summarized in Table 2. Compared to inference-only reasoning, DPO yields a 5.7% improvement

on the in-domain dataset 2WikiMQA, while improving performance on out-of-domain multi-hop QA benchmarks such as HotpotQA, as well as the short-form QA dataset SimpleQA.

Furthermore, we perform extensive experiments across different reasoning backbones and training data compositions, where CheckRLM also achieves strong performance on all evaluated datasets (see Appendix C.2 for complete results). These results demonstrate that our DPO data sampling and training pipeline effectively enhances CheckRLM across diverse reasoning models and datasets, exhibiting strong generality and robustness.

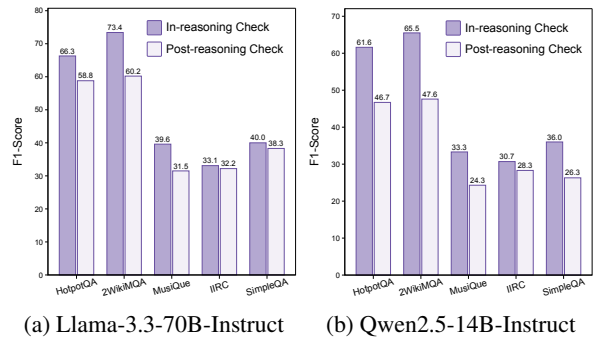


Figure 3: **Post-reasoning Check vs. In-reasoning Check**. We use QwQ-32B as the reasoning model, Llama-3.3-70B-Instruct and Qwen2.5-14B-Instruct as the recognition and correction model. The experiments are conducted in the 2WikiMQA dataset and use f1 as the metric. More results are in Appendix C.4.

5.3 Effectiveness of In-reasoning Knowledge Check in CheckRLM

CheckRLM introduces an intermediate intervention strategy that integrates knowledge claim recognition with consistency correction for in-reasoning check. Benchmarked against post-reasoning check, CheckRLM consistently outperforms across all datasets (Figure 3). The results confirm the superiority of in-reasoning intervention.

RLMs often introduce errors early in inference, which then propagate and amplify through subsequent steps. While post-reasoning check accesses the complete chain, its late correction can only address a subset of errors and fails to rectify already flawed reasoning trajectories. In contrast, CheckRLM intervenes dynamically during reasoning—correcting factual errors before they cascade and ensuring that each step proceeds under accurate knowledge constraints. This step-wise correction substantially improves the stability and reliability of the overall reasoning process.

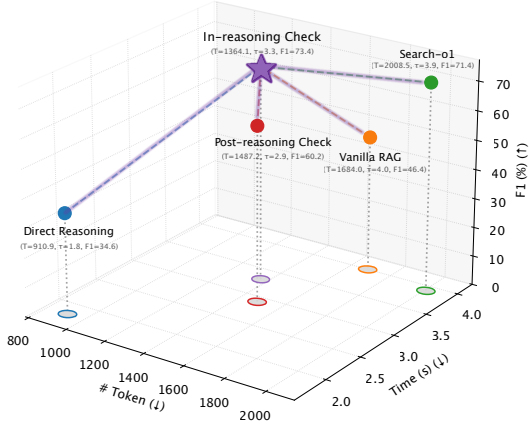


Figure 4: **Time, token consumption and performance of different methods.** We use QwQ-32B as the reasoning model, Llama-3.3-70B-Instruct as the recognition and correction model. The experiments are conducted in the 2WikiMQA dataset. More results are in Appendix C.7.

5.4 Analysis of Cost-Efficient Reasoning

In this experiment, we conduct a three-dimensional analysis that jointly considers token consumption, inference time, and task performance, aiming to provide a comprehensive evaluation of efficiency and performance.

As shown in Figure 4, Direct Reasoning has the lowest cost but limited performance. In contrast, Vanilla RAG and Search-o1 improve performance at substantially higher token and time costs. Notably, the in-reasoning check in CheckRLM attains a more favorable outcome across both performance and cost dimensions. Compared with Vanilla RAG and Search-o1, it significantly reduces token consumption and inference time while simultaneously achieving the best performance. This suggests that timely knowledge correction during reasoning enables the model to avoid unnecessary reasoning steps and redundant retrieval caused by error accumulation, improving efficiency without sacrificing accuracy.

Furthermore, the comparison between the in-reasoning check in CheckRLM and the post-reasoning check highlights the critical role of correction timing. Although post-reasoning check reduces computational cost relative to retrieval-heavy methods, it remains less effective than in-process check, as delayed correction cannot prevent erroneous knowledge from propagating throughout the reasoning chain. Overall, CheckRLM achieves the best performance while substantially reducing both inference time and token consumption, demonstrat-

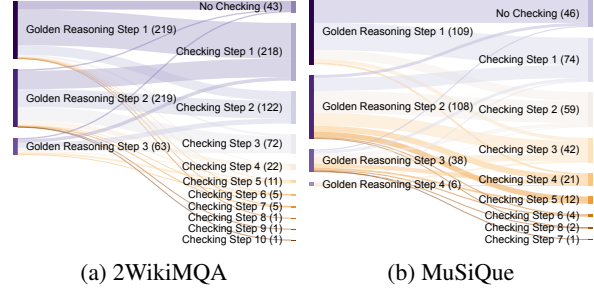


Figure 5: **Correspondence between Golden Reasoning Step and Checking Step.** We use QwQ-32B as the reasoning model, Llama-3.3-70B-Instruct as the recognition and correction model. The experiments are conducted in the 2WikiMQA and MuSiQue datasets.

ing clear superiority over all competing methods in terms of both efficiency and effectiveness.

5.5 Characteristics of Correction Behavior

This experiment aims to analyze when factual errors in multi-hop reasoning are detected and corrected by our method using the GPT-4o model, relative to the golden reasoning steps.

As depicted in Figure 5, Golden Reasoning Step k denotes the k -th hop in the golden reasoning chain, while Checking Step j indicates that our method successfully identifies and corrects the error at its j -th checking iteration. The observed trend shows that corrections predominantly occur at early checking steps (Steps 1–2), indicating effective early interception of factual errors rather than their propagation through the reasoning chain.

Despite its higher reasoning complexity, MuSiQue still exhibits a clear tendency toward early correction. While Checking Step 1 remains dominant, errors arising from later oracle steps are primarily resolved at intermediate stages (Steps 2–4), with deep checking being rare. This indicates that the method consistently corrects factual errors at early stages, even under more challenging multi-hop reasoning tasks. These results demonstrate that our method performs early, targeted, and efficient error correction, rather than relying on deep search or repeated post-hoc refinement.

6 Conclusion

In conclusion, we present **CheckRLM**, a framework that tackles factual errors in RLMs for knowledge-intensive tasks. CheckRLM enhances reasoning reliability by proactively recognizing and correcting factual inconsistencies during in-

ference via RAG. It extracts factual claims, locates errors against external knowledge, and performs precise, low-cost corrections. Experiments demonstrate that CheckRLM significantly outperforms existing baselines, mitigates error accumulation in long-horizon reasoning, and reduces temporal and computational costs, advancing towards more accurate and efficient knowledge-reliant reasoning.

Limitations

Although CheckRLM has successfully achieved real-time intervention of external knowledge for reasoning verification, this study still has certain limitations, mainly reflected in two aspects. First, the current mechanism focuses on utilizing pure textual knowledge sources and has not yet been extended to the collaborative verification of multimodal information. Second, the framework is currently optimized only for a single knowledge base and does not explore the complex coordination and conflict resolution mechanisms across multiple heterogeneous knowledge bases. Future research could delve into the directions of multimodal fusion and dynamic integration of multi-source knowledge to build a more general and robust reasoning verification framework.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62276026) and the Fundamental Research Funds for the Central Universities (No. 2253500001). This work is supported by the AI9Stars community.

References

- Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection.
- Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.
- James Ferguson, Matt Gardner, Hannaneh Hajishirzi, Tushar Khot, and Pradeep Dasigi. 2020. *IIRC: A dataset of incomplete information reading comprehension questions*. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1137–1147, Online. Association for Computational Linguistics.
- Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2024. *GPTScore: Evaluate as you desire*. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6556–6576, Mexico City, Mexico. Association for Computational Linguistics.
- Yunfan Gao, Yun Xiong, Yijie Zhong, Yuxi Bi, Ming Xue, and Haofen Wang. 2025. Synergizing rag and reasoning: A systematic review. *arXiv preprint arXiv:2504.15909*.
- Xinyan Guan, Jiali Zeng, Fandong Meng, Chunlei Xin, Yaojie Lu, Hongyu Lin, Xianpei Han, Le Sun, and Jie Zhou. 2026. *DeepRAG: Thinking to retrieve step by step for large language models*. In *The Fourteenth International Conference on Learning Representations*.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. *DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning*. *Nature*, 645(8081):633–638. TLDR: A new artificial intelligence model, DeepSeek-R1, is introduced, demonstrating that the reasoning abilities of large language models can be incentivized through pure reinforcement learning, removing the need for human-annotated demonstrations.
- Yancheng He, Shilong Li, Jiaheng Liu, Weixun Wang, Xingyuan Bu, Ge Zhang, Zy Peng, Zhaoxiang Zhang, Zhicheng Zheng, Wenbo Su, and 1 others. 2025. Can large language models detect errors in long chain-of-thought reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18468–18489.
- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. *Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps*. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Qisheng Hu, Quanyu Long, and Wenya Wang. 2025. Coordinating search-informed reasoning and reasoning-guided search in claim verification. *arXiv preprint arXiv:2506.07528*.
- Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard

- Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.
- 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)*, pages 7036–7050, Mexico City, Mexico. Association for Computational Linguistics.
- Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.
- Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. 2025. [Training language models to self-correct via reinforcement learning](#). In *The Thirteenth International Conference on Learning Representations*.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.
- Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. [Search-o1: Agentic search-enhanced large reasoning models](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438, Suzhou, China. Association for Computational Linguistics.
- Xinzhe Li. 2025. A survey on llm test-time compute via search: Tasks, llm profiling, search algorithms, and relevant frameworks. *arXiv preprint arXiv:2501.10069*.
- Yushen Li, Jinpeng Wang, Tao Dai, Jieming Zhu, Jun Yuan, Rui Zhang, and Shu-Tao Xia. 2024. Rat: Retrieval-augmented transformer for click-through rate prediction. In *Companion Proceedings of the ACM Web Conference 2024*, pages 867–870.
- Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. 2023. Deductive verification of chain-of-thought reasoning. *Advances in Neural Information Processing Systems*, 36:36407–36433.
- Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. [G-eval: NLG evaluation using gpt-4 with better human alignment](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.
- OpenAI. 2024. [Learning to reason with llms, september 2024](#).
- Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, and 1 others. 2021. Kilt: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.
- Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.
- Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. 2025. Zerosearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*.
- Qwen Team. 2025. [Qwq-32b: Embracing the power of reinforcement learning](#).
- Harsh Trivedi, Niranjana Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.
- Harsh Trivedi, Niranjana Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 10014–10037.

- Gladys Tyen, Hassan Mansoor, Victor Cărbune, Yuanzhu Peter Chen, and Tony Mak. 2024. LLMs cannot find reasoning errors, but can correct them given the error location. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13894–13908.
- Ruobing Wang, Qingfei Zhao, Yukun Yan, Daren Zha, Yuxuan Chen, Shi Yu, Zhenghao Liu, Yixuan Wang, Shuo Wang, Xu Han, Zhiyuan Liu, and Maosong Sun. 2025. **DeepNote: Note-centric deep retrieval-augmented generation**. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 19688–19715, Suzhou, China. Association for Computational Linguistics.
- Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, and 1 others. 2024. Searching for best practices in retrieval-augmented generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17716–17736.
- Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575.
- Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. **C-pack: Packaged resources to advance general chinese embedding**. *Preprint*, arXiv:2309.07597.
- Guangzhi Xiong, Qiao Jin, Xiao Wang, Yin Fang, Haolin Liu, Yifan Yang, Fangyuan Chen, Zhixing Song, Dengyu Wang, Minjia Zhang, and 1 others. 2025. Rag-gym: Optimizing reasoning and search agents with process supervision. *arXiv preprint arXiv:2502.13957*.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- 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, Jiayi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.
- Zijun Yao, Yantao Liu, Yanxu Chen, Jianhui Chen, Junfeng Fang, Lei Hou, Juanzi Li, and Tat-Seng Chua. 2025. Are reasoning models more prone to hallucination? *arXiv preprint arXiv:2505.23646*.
- Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H. Chi, and Denny Zhou. 2024. **Large language models as analogical reasoners**. In *International Conference on Learning Representations*, volume 2024, pages 17019–17045.
- Duzhen Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhijiang Guo, Le Song, and Cheng-Lin Liu. 2026. **From System 1 to System 2: A Survey of Reasoning Large Language Models**. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 48(03):3335–3354.
- Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. **Automatic chain of thought prompting in large language models**. In *The Eleventh International Conference on Learning Representations*.
- Qingfei Zhao, Ruobing Wang, Yukuo Cen, Daren Zha, Shicheng Tan, Yuxiao Dong, and Jie Tang. 2024. **LongRAG: A dual-perspective retrieval-augmented generation paradigm for long-context question answering**. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22600–22632, Miami, Florida, USA. Association for Computational Linguistics.
- Qingfei Zhao, Ruobing Wang, Dingling Xu, Daren Zha, and Limin Liu. 2025. R-search: Empowering LLM reasoning with search via multi-reward reinforcement learning. *arXiv preprint arXiv:2506.04185*.
- 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*.

Kunlun Zhu, Yifan Luo, Dingling Xu, Yukun Yan, Zhenghao Liu, Shi Yu, Ruobing Wang, Shuo Wang, Yishan Li, Nan Zhang, Xu Han, Zhiyuan Liu, and Maosong Sun. 2025. [RAGEval: Scenario specific RAG evaluation dataset generation framework](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8520–8544, Vienna, Austria. Association for Computational Linguistics.

Appendix

A	Details of DPO Training Data Construction	13
A.1	DPO Training Data for Knowledge Claim Recognition	13
A.2	DPO Training Data for Knowledge Coherence Correction	13
B	Prompt Details	13
B.1	Direct Reasoning Prompt	14
B.2	Vanilla RAG Prompt	14
B.3	CheckRLM Prompt	14
B.4	Correction Behavior Prompt	14
B.5	DPO Evaluation Prompt	14
C	Additional Experimental Results	15
C.1	Overall Performance of CheckRLM	15
C.2	Knowledge Recognition and Correction Optimization	15
C.3	Constrained Ablation Study	15
C.4	Post-reasoning Check vs. In-reasoning Check	15
C.5	Effect of the Retrieved Summary Injection on Reasoning	15
C.6	Effect of Different Retrievers	16
C.7	Cost-Efficient Reasoning Results	16
C.8	Correction Step Distribution	16
D	Case Study	16

A Details of DPO Training Data Construction

We randomly select 2500 samples from the 2WikiMQA training set to build our DPO training dataset \mathcal{D}_{DPO} , which comprises two subsets: training data for knowledge claim recognition \mathcal{D}_{KCR} and training data for knowledge coherence correction \mathcal{D}_{KCC} . The statistics of all DPO training data are presented in Table 3.

A.1 DPO Training Data for Knowledge Claim Recognition

Observing that knowledge corrections primarily occur in the initial steps of the reasoning process, we only sample data from the first three reasoning chain units to efficiently obtain high-quality data. For a given sample, we input the original question q , the current reasoning chain unit r_t , and the recognition instruction template Instruct_r into \mathcal{M}_{rec} to sample a set of knowledge claims y_t^{claim} . To balance diversity and accuracy in the

	\mathcal{D}_{KCR}	\mathcal{D}_{KCC}	\mathcal{D}_{DPO}
# Sample	2351	2960	5311

Table 3: Statistics of DPO Training Data.

generated samples, we set different decoding hyperparameters, namely temperature and top_ p , to specific values of 0.1, 0.5, and 0.9. They control the generation of a candidate answer set $\mathcal{Y}_t^{\text{claim}} = \{y_{t_1}^{\text{claim}}, y_{t_2}^{\text{claim}}, \dots, y_{t_n}^{\text{claim}}\}$, where $n \in \mathbb{N}^+$ denotes the number of candidate answers. Recent studies have demonstrated that annotations from advanced models can achieve high alignment with human judgments (Liu et al., 2023; Fu et al., 2024). We utilize GPT-4o-mini to annotate positive and negative examples $y_t^{\text{claim}(+)}$ and $y_t^{\text{claim}(-)}$ within the candidate answer set. The resulting training data \mathcal{D}_{KCR} can be represented as $\{x_t, y_t^{\text{claim}(+)}, y_t^{\text{claim}(-)}\}$.

A.2 DPO Training Data for Knowledge Coherence Correction

For training data \mathcal{D}_{KCC} , the settings for decoding hyperparameters remain the same as above. Since the quality of retrieved documents is positively correlated with the quality of knowledge claims, we use the positive example $y_t^{\text{claim}(+)}$ for document retrieval. The retrieved document set \mathcal{D}_t^+ , the current reasoning chain unit r_t , and the correction instruction template Instruct_c are then fed into \mathcal{M}_{cor} to generate a candidate answer set $\tilde{\mathcal{R}}_t = \{\tilde{r}_{t_1}, \tilde{r}_{t_2}, \dots, \tilde{r}_{t_n}\}$, where $n \in \mathbb{N}^+$ is the number of candidate answers. Similarly, we employ GPT-4o-mini to annotate positive and negative examples \tilde{r}_t^+ and \tilde{r}_t^- and construct the DPO training data for knowledge coherence correction, denoted as $\{x_t, \tilde{r}_t^+, \tilde{r}_t^-\}$. We then use the positive example \tilde{r}_t^+ to form the reasoning chain \mathcal{R} of the RLM to generate the next reasoning chain unit r_{t+1} .

B Prompt Details

This section presents the prompt templates employed for both inference and DPO training. For inference, we provide prompts for Direct Reasoning, Vanilla RAG, knowledge claim recognition, knowledge coherence correction and correction behavior analysis. In particular, the reasoning prompt used in CheckRLM is identical to the direct reasoning. For DPO training, we supply corresponding prompts for evaluating knowledge claim recognition and knowledge coherence correction.

B.1 Direct Reasoning Prompt

Direct Reasoning Prompt

Please reason step by step, and put your final answer within `\boxed{ }`. Only give me the answer and do not output any other words.

Question: {question}

B.2 Vanilla RAG Prompt

Vanilla RAG Prompt

Answer the question based on the given passages. Please reason step by step, and put your final answer within `\boxed{ }`. Only give me the answer and do not output any other words.

The following are given passages:

{refs}

Question: {question}

B.3 CheckRLM Prompt

Knowledge Claim Recognition Prompt

Task: Extract only explicit factual claims from the Reasoning Process that are helpful to answering the Question. The Reasoning Process contains factual information useful for addressing the Question. Your task is to identify, extract, and summarize these relevant facts into clear factual claims. The factual claims must be based on the Reasoning Process; NEVER add any new content by yourself. Try to avoid two factual claims that have similar meaning. Output is a list of strings. The output format is as follows:

["factual claim 1", "factual claim 2", ...]

Question: {question}

Reasoning Process: {reasoning}

Factual Claim List:

Knowledge Coherence Correction Prompt

Task: Based on the retrieved documents, check and correct factual errors in the Reasoning Process. You will receive several retrieved documents and the Reasoning Process. The retrieved documents may contain information relevant to the Reasoning Process. Your task is to filter the relevant information from the retrieved documents and verify the correctness of the factual information in the Reasoning Process. Follow these rules:

1. If the factual information in the Reasoning Process is correct, only output the original Reasoning Process.
2. If the factual information in the Reasoning Process is incorrect, make the minimal necessary corrections to fix the error without altering the structure or flow of the Reasoning Process.
3. If the retrieved documents do not contain any relevant information, only output the original Reasoning Process. NEVER add any supplementary information from the retrieved documents. Only correct factual errors when necessary. Only give me your modified reasoning process and do not output any other words.

Retrieved documents: {refs}

Reasoning Process: {reasoning}

Provide your modified reasoning process:

B.4 Correction Behavior Prompt

Correction Behavior Prompt

Task: Label Effective Correction Steps

You are a data annotation expert tasked with identifying which refinement steps successfully introduce key content from the correct reasoning chain.

Requirements:

1. Examine the "original think" and "refine think" content of each step one by one.
2. If "original think" content contains the key step from the correct reasoning chain, output the correct reasoning step number and -1.
3. If the "refine think" content contains any key step from the correct reasoning chain, output the correct reasoning step number and the step number of the modified content.
4. If multiple steps contain the same key step, only record the earliest occurrence.

Output Rules:

1. Output format: [{"Correct Reasoning Step Number": "First Modified Number"}]
2. Do not include any explanations or additional text. Only output the list.

Input:

Question: {query}

Correct Reasoning Chains: {golden_reasoning_steps}

Modification Records: {think_refine_detail}

B.5 DPO Evaluation Prompt

Knowledge Claim Recognition Evaluation Prompt

Task: Evaluate factual claim lists generated based on a Reasoning Process and Question. Your task is to select the best factual claim list and the worst factual claim list according to Quality criteria strictly. Quality criteria refer to:

1. Relevance: each factual claim in the factual claim list is relevant to the Question and Reasoning Process, note that if there is no factual claim in Reasoning Process, just output [] but not similar sentences such as "there is no related information" or "the reasoning process does not contain any factual claim";
2. Specificity: each factual claim in the factual claim list is clear, avoiding unclear pronoun;
3. No redundancy: The correct factual claim list format is just one list of strings, no multiple lists or explanatory Notes.

Output Rules:

1. JSON format: {"best_id": <id of the factual claim list that fully meets all Quality criteria above>, "worst_id": <id of the factual claim list that violates the most Quality criteria>};
2. If there is a minimal quality difference between the best factual claim list and the worst factual claim list or there is no worst factual claim, assign the same id to both best_id and worst_id;
3. If no factual claim list meets all the Quality criteria above, return an empty object {}
4. Do not include any explanations or additional text.

Question: {question}

Reasoning Process: {reasoning}

Generated factual claim lists: {check_responses}

Knowledge Coherence Correction Evaluation Prompt

Task: Evaluate refined reasoning processes based on quality criteria strictly and identify the best and the worst refined reasoning processes. Quality criteria refer to:

1. Structural Integrity: strictly preserves the original reasoning process structure. Do not add any other supplement information at the end of the original reasoning process, even if the supplement information is right. The refined reasoning process contains negative sentences such as “the retrieved documents do not contain related information” is the worst;
2. Precise Corrections: Only modifies factually incorrect content verified by retrieved documents.
3. Conciseness: No redundant text in the refined reasoning process, including unnecessary prefixes, explanatory Notes, and continuation after the original reasoning process, such as “According to the retrieved documents”.

Output Rules:

1. JSON format: {"best_id": <id of the refined reasoning process that fully meets all Evaluation Criteria>, "worst_id": <id of the refined reasoning process that violates the most Quality criteria>}
 2. If there is a minimal quality difference between the best reasoning process and the worst reasoning process or there is no worst refined reasoning process, assign the same id to both best_id and worst_id
 3. If no refined reasoning process meets all Evaluation Criteria above, return an empty object {}
 4. Do not include any explanations or additional text.
- Retrieved documents: {refs}
Original reasoning process: {think}
Generated refined reasoning processes: {refine_responses}

C Additional Experimental Results

C.1 Overall Performance of CheckRLM

The Table 4 presents overall results of various RLMs and recognition and correction models. By combining reasoning models and recognition and correction models of varying parameter sizes, we find that CheckRLM consistently achieves better performance than both Direct Reasoning and Vanilla RAG across all datasets, demonstrating the generality and robustness of CheckRLM.

C.2 Knowledge Recognition and Correction Optimization

The Table 5 presents all results of CheckRLM with DPO training. Using the Qwen2.5-14B-Instruct model as the backbone, we conduct experiments across different reasoning models and datasets. The results show that DPO training outperforms Inference Only in almost all cases, demonstrating the effectiveness and stability of our constructed training data pipeline and training.

To further analyze the impact of different data compositions, we conduct additional experiments (Table 6) and find that DPO training on the com-

posed data $\mathcal{D}_{\text{KCR}} + \mathcal{D}_{\text{KCC}}$ outperforms training on single-type data, demonstrating the effectiveness of both \mathcal{D}_{KCR} and \mathcal{D}_{KCC} . Furthermore, the performance improvement brought by \mathcal{D}_{KCR} is more pronounced in most cases, which can be attributed to the fact that higher-quality factual claims enable the correction model to rectify factual errors more accurately.

C.3 Constrained Ablation Study

In the constrained ablation study, given the high interdependence between the In-Process Knowledge Claim Recognition and Localized Knowledge Coherence Correction modules that prevent their complete removal, we apply the simplest versions of both modules and conduct experiments across various models. The experimental results, as shown in the Table 7, demonstrate the importance of both modules within the CheckRLM framework.

C.4 Post-reasoning Check vs. In-reasoning Check

The Table 9 compares the complete results of Post-reasoning Check and In-reasoning Check under various combinations of RLMs and recognition and correction models. The In-reasoning Check results outperform the Post-reasoning Check method across all experimental settings, demonstrating the superiority of CheckRLM.

C.5 Effect of the Retrieved Summary Injection on Reasoning

The goal of CheckRLM correction is to correct errors in the reasoning chain with minimal cost, while preserving the original distribution of the reasoning model as much as possible. Our observations indicate that the reasoning model itself exhibits a clear and coherent problem-solving approach, and that incorporating additional information may interfere with its inherent reasoning logic. To investigate this phenomenon, we design a comparative experiment that injects summarized retrieved information into the reasoning chain during error correction.

The experimental results are presented in the Table 8. We observe that adding more relevant information alongside factual error correction leads to a decline in performance. Additionally, we find that when no relevant documents are retrieved during the reasoning process, the correction model tends to append statements such as “*The retrieved documents do not provide any useful information...*” to the summary. This can mislead the reasoning

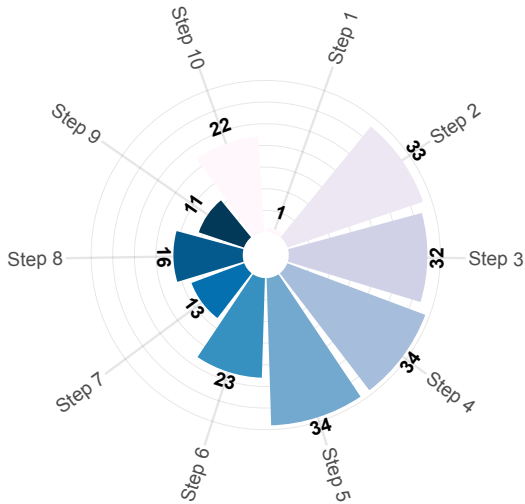


Figure 6: Correction step distribution in 2WikiMQA.

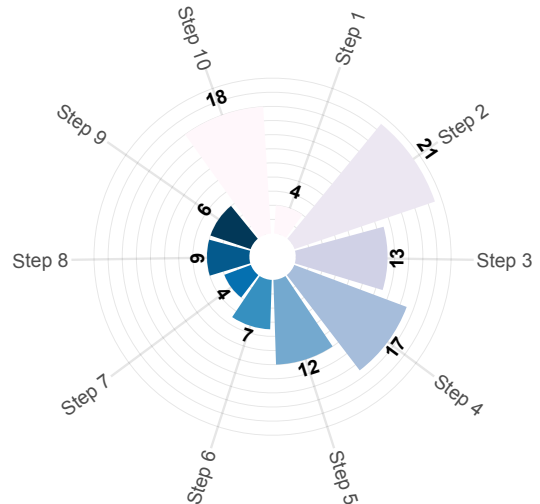


Figure 7: Correction step distribution in MuSiQue.

model into assuming that the problem is unsolvable, causing it to halt exploration and directly output an incorrect answer with no response, thereby reducing system robustness. Therefore, it is more effective for CheckRLM to minimally revise the reasoning chain rather than injecting summarized retrieved information directly into it.

C.6 Effect of Different Retrievers

The Table 10 presents the impact of two different retrievers, the sparse retriever BM25 and the dense retriever bge-large-en-v1.5, on the CheckRLM framework. It can be observed that bge-large-en-v1.5 yields better performance than BM25. Furthermore, CheckRLM significantly outperforms both Direct Reasoning and Vanilla RAG across both retrieval settings, demonstrating the robustness of our proposed method.

C.7 Cost-Efficient Reasoning Results

Table 11 and Table 12 present more results of Cost-Efficient Reasoning.

C.8 Correction Step Distribution

We use QwQ-32B as the reasoning model and Llama-3.3-70B-Instruct as the recognition and correction model. Figures 6 and 7 present the distribution of correction steps for CheckRLM on 2WikiMQA and MuSiQue.

Across both datasets, these distributions exhibit a strong concentration on early checking steps, particularly Step 2-5, indicating that most factual inconsistencies are corrected promptly after they are introduced. This pattern suggests that CheckRLM effectively intercepts errors at an early stage of

the reasoning process, preventing erroneous knowledge from being propagated to subsequent reasoning steps. Notably, deeper correction steps occur only rarely, even on MuSiQue, which involves more complex and intertwined multi-hop reasoning. This observation implies that the checking mechanism does not degenerate into excessive late-stage corrections, but instead resolves most errors before they accumulate and compound. Overall, these results highlight that the effectiveness of CheckRLM stems from early and timely correction, which plays a critical role in mitigating errors accumulated in long-step reasoning.

D Case Study

Tables 13, 14, 15 present three cases based on the same question, comparing the outputs of Direct Reasoning, CheckRLM with Post-reasoning Check, and CheckRLM with In-reasoning Check.

Given the question “*When was the director of the film Slap Her... She’s French born?*”, the correct reasoning process should first identify the director’s name and then infer the birth date. RLM Direct Reasoning incorrectly identifies the director’s name, and this error propagates to the subsequent step of determining the birth date, leading to an incorrect final answer. Both Vanilla RAG and CheckRLM Post-reasoning Check correctly identify the director’s name; however, they also produce an incorrect final answer due to the absence of retrieved document segments with the director’s birth date.

In contrast, CheckRLM via In-reasoning Check corrects the director’s name from “*David Zucker*” to the correct “*Melanie Mayron*” in Step 1, and

it then revises Melanie Mayron's birth date from the erroneous "*July 21, 1968*" to the correct "*October 20, 1952*" in Step 2, ultimately arriving at the right answer. This illustrates the necessity of timely error correction and demonstrates the effectiveness of CheckRLM and its In-reasoning Check strategy. Moreover, we observe that the correction model can supplement relevant knowledge pertinent to the question sometimes while correcting erroneous information, significantly streamlining the reasoning process and enabling more efficient inference.

Method	Multi-hop								Short-form		Avg.
	HotpotQA		2WikiMQA		MuSiQue		IIRC		SimpleQA		
	f1	em	f1	em	f1	em	f1	em	f1	em	
<i>QwQ-32B</i>											
Direct Reasoning	38.4	29.0	34.6	28.4	18.5	7.4	24.8	21.4	10.5	4.6	21.8
Vanilla RAG	52.7	42.2	46.4	42.6	19.3	10.0	25.0	22.0	31.4	24.2	31.6
CheckRLM											
- Check@Qwen-3-8B	59.9	47.4	59.9	50.6	29.0	17.8	29.0	25.4	33.4	25.2	37.8
- Check@Qwen-2.5-14B	61.6	48.6	65.5	54.8	33.3	21.2	30.7	26.0	36.0	27.3	40.5
- Check@Qwen-2.5-32B	59.4	46.6	66.3	56.6	33.1	19.8	30.9	26.2	34.0	25.2	39.8
- Check@Llama-3.3-70B	66.3	52.6	73.4	62.0	39.6	27.2	33.1	29.0	40.0	30.4	45.4
<i>Qwen3-32B</i>											
Direct Reasoning	36.4	27.6	34.0	29.4	16.6	6.6	23.0	20.2	9.5	2.4	20.6
Vanilla RAG	51.2	41.0	44.5	41.8	19.7	10.4	22.5	19.8	29.3	22.8	30.3
CheckRLM											
- Check@Qwen-2.5-14B	57.0	46.8	62.9	50.2	29.9	20.0	26.5	23.2	34.2	26.2	37.7
- Check@Llama-3.3-70B	64.2	52.0	70.8	60.8	35.6	25.4	31.6	27.8	39.1	30.6	43.8
<i>Qwen3-8B</i>											
Direct Reasoning	29.3	22.2	31.1	26.6	12.9	5.2	21.0	18.2	7.7	2.8	17.7
Vanilla RAG	44.5	35.2	40.9	38.4	13.8	6.0	19.8	16.8	29.4	23.0	26.8
CheckRLM											
- Check@Qwen-2.5-32B	56.3	45.0	60.0	50.0	28.2	18.0	25.0	22.2	33.0	26.2	36.4
- Check@Llama-3.3-70B	61.9	49.6	69.7	58.6	35.0	24.0	29.7	25.4	35.7	28.6	41.8

Table 4: Comprehensive benchmark results (%) using different RLMs paired with various recognition and correction models.

Method	Multi-hop								Short-form	
	HotpotQA		2WikiMQA		MuSiQue		IIRC		SimpleQA	
	f1	em	f1	em	f1	em	f1	em	f1	em
<i>Reasoning: QwQ-32B & Checking: Qwen-2.5-14B</i>										
Vanilla RAG	52.7	42.2	46.4	42.6	19.3	10.0	25.0	22.0	31.4	24.2
Inference Only	61.6	48.6	65.5	54.8	33.3	21.2	30.7	26.0	36.0	27.3
DPO Training	63.2	50.0	71.2	58.4	36.2	22.0	32.2	28.0	36.9	28.0
<i>Reasoning: Qwen3-32B & Checking: Qwen-2.5-14B</i>										
Vanilla RAG	51.2	41.0	44.5	41.8	19.7	10.4	22.5	19.8	29.3	22.8
Inference Only	57.0	46.8	62.9	50.2	29.9	20.0	26.5	23.2	34.2	26.2
DPO Training	61.3	48.6	65.3	54.8	32.7	20.2	31.0	26.6	36.1	28.8
<i>Reasoning: DeepSeek-R1-Distill-Llama-70B & Checking: Qwen-2.5-14B</i>										
Vanilla RAG	56.1	43.0	49.4	44.4	25.1	14.6	26.1	22.6	32.0	25.0
Inference Only	60.9	47.4	64.1	53.4	32.3	20.8	29.0	23.0	35.9	27.9
DPO Training	62.7	49.2	65.7	54.4	31.7	20.0	31.2	26.0	36.8	27.8

Table 5: Results (%) of performance with DPO training.

Method	Multi-hop								Short-form	
	HotpotQA		2WikiMQA		MuSiQue		IIRC		SimpleQA	
	f1	em	f1	em	f1	em	f1	em	f1	em
<i>Reasoning: QwQ-32B & Checking: Qwen-2.5-14B</i>										
\mathcal{D}_{KCR} Only	59.9	48.0	67.9	56.2	33.7	21.2	31.2	27.0	36.2	27.4
\mathcal{D}_{KCC} Only	60.3	47.6	68.9	56.0	33.8	20.6	29.9	25.6	34.1	24.0
$\mathcal{D}_{\text{KCR}}+\mathcal{D}_{\text{KCC}}$	63.2	50.0	71.2	58.4	36.2	22.0	32.2	28.0	36.9	28.0

Table 6: Results (%) of performance with different training data compositions.

Method	Multi-hop								Short-form	
	HotpotQA		2WikiMQA		MuSiQue		IIRC		SimpleQA	
	f1	em	f1	em	f1	em	f1	em	f1	em
<i>Reasoning: Qwen3-8B & Checking: Qwen2.5-32B</i>										
CheckRLM	56.3	45.0	60.0	50.0	28.2	18.0	25.0	22.2	33.0	26.2
- w/o Verification	45.3	36.8	54.4	47.0	20.1	10.8	24.2	20.8	24.3	17.0
- w/o Refinement	50.2	39.8	56.0	48.2	26.2	15.2	24.1	20.4	32.3	24.8
<i>Reasoning: QwQ-32B & Checking: Qwen2.5-14B</i>										
CheckRLM	61.6	48.6	65.5	54.8	33.3	21.2	30.7	26.0	36.0	27.3
- w/o Verification	52.2	40.2	62.0	54.2	30.2	18.6	29.1	25.2	30.6	21.6
- w/o Refinement	58.0	45.4	64.2	53.0	29.7	18.6	28.2	23.0	33.3	25.8
<i>Reasoning: QwQ-32B & Checking: Llama-3.3-70B</i>										
CheckRLM	66.3	52.6	73.4	62.0	39.6	27.2	33.1	29.0	40.0	30.4
- w/o Verification	62.9	50.4	72.6	60.8	39.4	26.0	33.0	28.6	37.7	28.4
- w/o Refinement	64.9	52.4	68.0	56.8	37.5	23.6	32.7	28.4	36.8	29.0

Table 7: All results (%) of constrained ablation study.

Method	Multi-hop								Short-form	
	HotpotQA		2WikiMQA		MuSiQue		IIRC		SimpleQA	
	f1	em	f1	em	f1	em	f1	em	f1	em
<i>Reasoning: QwQ-32B & Checking: Llama-3.3-70B</i>										
CheckRLM	66.3	52.6	73.4	62.0	39.6	27.2	33.1	29.0	40.0	30.4
- w/ Summary	66.2	52.0	71.5	59.6	39.8	25.2	33.0	28.0	38.5	29.8

Table 8: Comparison (%) of CheckRLM with and without retrieved summary injection. “w/ Summary” denotes injecting summarized retrieved information into the reasoning chain during correction.

Method	Multi-hop								Short-form	
	HotpotQA		2WikiMQA		MuSiQue		IIRC		SimpleQA	
	f1	em	f1	em	f1	em	f1	em	f1	em
<i>Reasoning: Qwen3-8B & Checking: Qwen2.5-32B</i>										
Post-reasoning Check	46.8	36.8	49.1	42.6	17.9	8.2	24.3	20.4	30.7	22.6
In-reasoning Check	56.3	45.0	60.0	50.0	28.2	18.0	25.0	22.2	33.0	26.2
% improv.	9.5	8.2	10.9	7.4	10.3	9.8	0.7	1.8	2.3	3.6
<i>Reasoning: QwQ-32B & Checking: Qwen2.5-14B</i>										
Post-reasoning Check	46.7	35.8	47.6	39.6	24.3	13.0	28.3	23.6	26.3	18.6
In-reasoning Check	61.6	48.6	65.5	54.8	33.3	21.2	30.7	26.0	36.0	27.3
% improv.	14.9	12.8	17.9	15.2	9.0	8.2	2.4	2.4	9.7	8.7
<i>Reasoning: QwQ-32B & Checking: Llama-3.3-70B</i>										
Post-reasoning Check	58.8	47.0	60.2	52.2	31.5	19.4	32.2	27.6	38.3	29.0
In-reasoning Check	66.3	52.6	73.4	62.0	39.6	27.2	33.1	29.0	40.0	30.4
% improv.	7.5	5.6	13.2	9.8	8.1	7.8	0.9	1.4	1.7	1.4

Table 9: **Post-reasoning Check vs. In-reasoning Check.** We compare the performance of CheckRLM with Post-reasoning Check and In-reasoning Check.

Method	Multi-hop							
	HotpotQA		2WikiMQA		MuSiQue		IIRC	
	f1	em	f1	em	f1	em	f1	em
<i>QwQ-32B</i>								
Direct Reasoning	38.4	29.0	34.6	28.4	18.5	7.4	24.8	21.4
Vanilla RAG	52.7	42.2	46.4	42.6	19.3	10.0	25.0	22.0
CheckRLM								
BM25	66.3	52.6	73.4	62.0	39.6	27.2	33.1	29.0
bge-large-en-v1.5	65.2	51.8	74.3	63.8	40.1	26.0	35.7	31.0
<i>Qwen3-32B</i>								
Direct Reasoning	36.4	27.6	34.0	29.4	16.6	6.6	23.0	20.2
Vanilla RAG	51.2	41.0	44.5	41.8	19.7	10.4	22.5	19.8
CheckRLM								
BM25	64.2	52.0	70.8	60.8	35.6	25.4	31.6	27.8
bge-large-en-v1.5	67.7	54.2	74.2	63.4	39.6	26.0	34.3	29.8
<i>Qwen3-8B</i>								
Direct Reasoning	29.3	22.2	31.1	26.6	12.9	5.2	21.0	18.2
Vanilla RAG	44.5	35.2	40.9	38.4	13.8	6.0	19.8	16.8
CheckRLM								
BM25	61.9	49.6	69.7	58.6	35.0	24.0	29.7	25.4
bge-large-en-v1.5	64.9	51.8	74.7	64.6	36.1	24.8	35.2	30.6

Table 10: **Results (%) of different retrievers.** We present the results of CheckRLM using Llama-3.3-70B-Instruct as the recognition and correction model.

Method	# Tokens (↓)	Time (s) (↓)	f1 (%) (↑)
Direct Reasoning	910.9	1.8	34.6
Vanilla RAG	1684.0	4.0	46.4
Search-o1	2008.5	3.9	71.4
Post-reasoning Check	1487.2	2.9	60.2
In-reasoning Check	1364.1	3.3	73.4

Table 11: **Cost-Efficient reasoning results (%)**. We conduct experiments on the 2WikiMQA dataset, using QwQ-32B as the reasoning model and Llama-3.3-70B-Instruct as the recognition and correction model.

Method	# Tokens (↓)	Time (s) (↓)	f1 (%) (↑)
Direct Reasoning	910.9	1.8	34.6
Vanilla RAG	1684.0	4.0	46.4
Search-o1	2008.5	3.9	71.4
Post-reasoning Check	1496.7	4.2	47.6
In-reasoning Check	1082.8	2.0	65.5

Table 12: **Cost-Efficient reasoning results (%)**. We conduct experiments on the 2WikiMQA dataset, using QwQ-32B as the reasoning model and Qwen2.5-14B-Instruct as the recognition and correction model.

Query: When was the director of film Slap Her... She's French born?

Golden Reasoning Path: Slap Her... She's French director **Melanie Mayron** -> Melanie Mayron date of birth **October 20, 1952**

Answer: October 20, 1952.

Direct Reasoning Output:

<think> Okay, I need to find out when the director of the film "Slap Her... She's French" was born. First, I should figure out who the director of that movie is. Let me recall... I think the movie is from the 1980s. The title sounds familiar. Maybe it's directed by someone like **Jim Abrahams**? Wait, Jim Abrahams was part of the team that did the Police Academy movies. Let me check that.

Alternatively, maybe it's **Mel Brooks**? No, Mel Brooks did Blazing Saddles and Young Frankenstein. Wait, Slap Her... She's French, that sounds like a cult movie. Let me think again. Oh, right, the director is Jim Abrahams. Wait, but I'm not 100% sure. Let me confirm. The movie came out in 1975, I believe. The director is Jim Abrahams, but I should make sure. Alternatively, maybe it's his brother, David, or one of the other brothers? Wait, Jim Abrahams, David Zucker, and Jerry Zucker? No, maybe not. Let me think again. The film "Slap Her... She's French" is a 1975 comedy. The director is indeed Jim Abrahams. Now, when was Jim Abrahams born?

Jim Abrahams was born on February 25, 1940. Wait, but I need to confirm that. Let me think. Jim Abrahams' birth date. I might be mixing up with his brother David. Wait, David Zucker was born in 1949. Hmm. Alternatively, maybe Jim Abrahams was born in 1940. Let me see.

Alternatively, perhaps I can recall that he was part of the team that did the Airplane! movies, but that's more the Zucker-Abrahams team. Wait, Jim Abrahams was born in 1940. Let me check in my mind. If the movie was 1975, and he was, say, in his mid-30s then, that would make his birth year around 1940. So, I think the correct answer is February 25, 1940. So the director, Jim Abrahams, was born on that date. I'll go with that. **</think>**

Answer: **February 25, 1940**

Vanilla RAG Output:

<think> Okay, I need to find out when the director of the film "Slap Her... She's French" was born. Let's look at the given passages.

First, the film's title is mentioned in two passages. Both say the film is directed by **Melanie Mayron** and starring Piper Perabo. The title of the film is "Slap Her... She's French," which was released in 2002. The question is about the director's birth year, so I need to find information related to Melanie Mayron.

Looking at the third passage about Jane McGregor, it states she was born in 1983 and lists her roles, including "Slap Her ... She's French" from 2002. However, Jane McGregor is an actress, not the director. The director is Melanie Mayron, whose details aren't provided in any of the given texts. The passages don't mention Melanie Mayron's birth date. The only birth year given here is Jane McGregor's, but that's irrelevant to the director. Since the information isn't present in the provided passages, **the answer can't be determined from the given data.** The user's question might be expecting me to realize that the passages don't include the director's birth year. **</think>**

Answer: **Unknown**

Table 13: **Case Study I.** This case illustrates the outputs of Direct Reasoning via RLM and vanilla RAG. We use QwQ-32B as the backbone model for reasoning.

Query: When was the director of film Slap Her... She's French born?

Golden Reasoning Path: Slap Her... She's French director **Melanie Mayron** -> Melanie Mayron date of birth **October 20, 1952**

Answer: October 20, 1952.

CheckRLM Post-reasoning Check Output:

Reasoning Output: <think> Okay, I need to find out when the director of the film "Slap Her... She's French" was born. First, I should figure out who the director of that movie is. Let me recall... I think the movie is from the 1980s. The title sounds familiar. Maybe it's directed by someone like Jim Abrahams? Wait, Jim Abrahams was part of the team that did the Police Academy movies. Let me check that.

Alternatively, maybe it's Mel Brooks? No, Mel Brooks did Blazing Saddles and Young Frankenstein. Wait, Slap Her... She's French, that sounds like a cult movie. Let me think again. Oh, right, the director is **Jim Abrahams**. Wait, but I'm not 100% sure. Let me confirm. The movie came out in 1975, I believe. The director was Jim Abrahams, who was part of the Three Amigos, along with David and Jerry. So Jim Abrahams is the director here. Now, when was he born?

Jim Abrahams was born on... I think he was born in the 1930s or 1940s. Let me recall. David and Jerry are his brothers, part of the comedy trio. Wait, Jim Abrahams was born on July 16, 1940. Let me see if that's correct. Alternatively, maybe 1941. Hmm, I might need to double-check. Wait, another way: if the movie was released in 1975, and he was in his 30s then, that would make his birth year around 1940-1945. So 1940 seems right. I think that's correct. So the answer should be **July 16, 1940**. </think>

Factual Claim:

1. Jim Abrahams is the director of the film 'Slap Her... She's French'.
2. Jim Abrahams was born on July 16, 1940.

Refine Output: <think> Okay, I need to find out when the director of the film "Slap Her... She's French" was born. First, I should figure out who the director of that movie is. Let me recall... I think the movie is from the 2000s. The title sounds familiar. Maybe it's directed by someone like Jim Abrahams? Wait, Jim Abrahams was part of the team that did the Police Academy movies. Let me check that.

Alternatively, maybe it's Mel Brooks? No, Mel Brooks did Blazing Saddles and Young Frankenstein. Wait, Slap Her... She's French, that sounds like a cult movie. Let me think again. Oh, right, the director is **Melanie Mayron**. Wait, but I'm not 100% sure. Let me confirm. The movie came out in 2002, I believe. The director was Melanie Mayron. Now, when was she born?

I do not have the exact birth date of Melanie Mayron from the provided documents.

However, based on the information that the movie was released in 2002 and considering she was likely in her early career or mid-career at that time, I would need to look up her birth year elsewhere. For the purpose of this reasoning process, I will not provide a specific birth date as it is not available in the given documents. </think>

Answer: **1965**

Table 14: **Case Study II.** This case illustrates the output of CheckRLM via Post-reasoning Check. We use QwQ-32B as the reasoning model and Llama-3.3-70B-Instruct as the recognition and correction model.

Query: When was the director of film Slap Her... She's French born?

Golden Reasoning Path: Slap Her... She's French director **Melanie Mayron** -> Melanie Mayron date of birth **October 20, 1952**

Answer: October 20, 1952.

CheckRLM In-reasoning Output:

Step 1:

Reasoning Output: <think> Okay, I need to find out when the director of the film "Slap Her... She's French" was born. First, I should figure out who the director of that movie is. Let me recall... I think the movie is from the 1980s. The title sounds familiar. Maybe it's **directed by someone like David Zucker**? Wait, the Zucker-Abrahams-Zucker team did a lot of comedies around that time. Let me confirm.

Factual Claim:

1. The movie 'Slap Her... She's French' is from the 1980s.
2. The Zucker-Abrahams-Zucker team directed many comedies around that time.

Refine Output: <think> Okay, I need to find out when the director of the film "Slap Her... She's French" was born. First, I should figure out who the director of that movie is. Let me recall... The movie is from 2002, not the 1980s. The title sounds familiar, but it's not directed by someone like David Zucker. **The movie is directed by Melanie Mayron.** Let me confirm Melanie Mayron's birth date.

Step 2:

Reasoning Output: Wait, I think **Melanie Mayron was born on July 21, 1968.** I should double-check that. Alternatively, maybe I'm mixing her up with someone else. Let me see... Another thought: sometimes people confuse the director with the cast. The film's cast includes actors like Adam Goldberg, but the director is definitely Melanie Mayron.

Factual Claim:

1. Melanie Mayron is the director of the film Slap Her... She's French.
2. Melanie Mayron was born on July 21, 1968.

Refine Output: Wait, I think **Melanie Mayron was born on October 20, 1952.** I should double-check that. Alternatively, maybe I'm mixing her up with someone else. Let me see... Another thought: sometimes people confuse the director with the cast. The film's cast includes actors like Piper Perabo, but the director is definitely Melanie Mayron.

Step 3:

Reasoning Output: So, Melanie Mayron's birth date is October 20, 1952. So the answer should be October 20, 1952. I'll make sure there's no conflicting info. Another check: if I remember correctly, she was born in 1952. Yeah, that seems right. So the director's birth date is 1952-10-20. </think>

Factual Claim:

1. Melanie Mayron's birth date is October 20, 1952.
2. The director of the film Slap Her... She's French is Melanie Mayron.

Refine Output: So, Melanie Mayron's birth date is October 20, 1952. So the answer should be October 20, 1952. I'll make sure there's no conflicting info. Another check: if I remember correctly, she was born in 1952. Yeah, that seems right. So the director's birth date is 1952-10-20. </think>

Answer: **October 20, 1952**

Table 15: **Case Study III.** This case illustrates the output of CheckRLM via In-reasoning Check. We use QwQ-32B as the reasoning model and Llama-3.3-70B-Instruct as the recognition and correction model.