LCR-RAG: Enhancing Logical Consistency in Retrieval-Augmented Generation via Neuro-symbolic Reinforcement Learning

Wenxiang Zheng[1], **Guo Tang**[1], **Shixin Jiang**[1], **Liangyu Huo**[3], **Xiyuan Zhang**[3], **Jian Xie**[3], **Ming Liu**[1,2*]
[1]Harbin Institute of Technology, Harbin, China
[2]Peng Cheng Laboratory, Shenzhen, China
[3]Du Xiaoman Technology, Beijing, China
{wxzheng, gtang, sxjiang, mliu}@ir.hit.edu.cn

Abstract

Retrieval-Augmented Generation (RAG) is widely used to ground large language models (LLMs) in external knowledge and improve factual accuracy. Prior work has explored iterative and self-reflective mechanisms to refine reasoning, but these approaches rely on internal model judgment and lack formally grounded, verifiable feedback. As a result, RAG systems may still produce logically inconsistent or contradictory answers in multi-step reasoning. In this paper, we propose LCR-RAG, a framework that integrates neuro-symbolic verification with reinforcement learning to explicitly optimize logical consistency. The core of our approach is a Logic-Consistency-driven Reward (LCR), which converts discrete logical signals—such as contradictions or incomplete inference chains—into a structured reward signal. This reward guides a PPO-based agent to iteratively rewrite queries and correct reasoning errors. Experiments on HotpotQA, ASQA, and TriviaQA show that LCR-RAG consistently outperforms strong RAG baselines, with ablation results indicating that the LCR mechanism is the primary source of improvement, even under noisy or conflicting retrieval conditions.

1 Introduction

The advent of Large Language Models (LLMs) has fundamentally reshaped the landscape of artificial intelligence, offering unprecedented capabilities in understanding and generating human-like text. A cornerstone of their practical application has been the development of Retrieval-Augmented Generation (RAG) frameworks (Lewis et al., 2020), which ground these models in external, verifiable knowledge sources. This paradigm has proven remarkably effective at mitigating factual hallucinations and enhancing the accuracy of answers in standard question-answering (QA) tasks. However, as the ambition of AI systems moves beyond simple fact retrieval towards complex, multi-step reasoning, the limitations of current RAG systems have become increasingly apparent (Kandpal et al., 2023). Many high-value real-world problems require not just the aggregation of disparate facts (Gao et al., 2023a), but a logically sound, step-by-step inference process to connect them. It is in these reasoning-intensive scenarios that existing systems exhibit a significant "Logic Deficit" (Cheng et al., 2025), often producing answers that, while factually grounded in individual retrieved documents, are internally contradictory or logically invalid when synthesized.

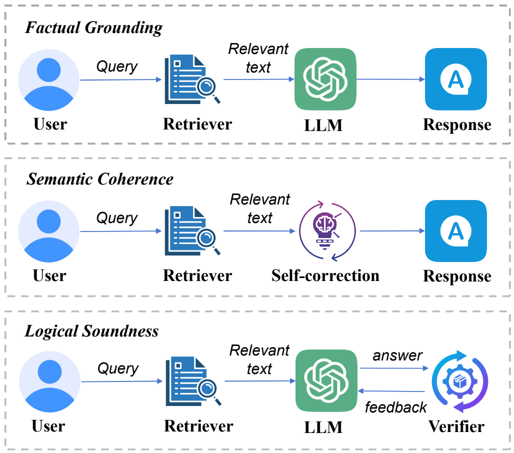

Figure 1: The RAG Reliability Pyramid, illustrating three ascending levels of trustworthiness: L1 Factual Grounding, L2 Semantic Coherence, and L3 Logical Soundness.

To systematically understand and address this challenge, we propose a "RAG Reliability Pyramid" (Gao et al., 2023b) as a conceptual model. This model delineates three ascending levels of trustworthiness for RAG systems. The foundational level, L1: Factual Grounding, is achieved by standard RAG, which ensures that generated statements are rooted in retrieved evidence. The next level, L2: Semantic Coherence, is addressed by

*Corresponding Author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 17900–17916
July 2-7, 2026 ©2026 Association for Computational Linguistics

advanced iterative frameworks such as IRCoT and Self-RAG (Asai et al., 2024). These systems employ Chain-of-Thought reasoning or internal self-reflection mechanisms to produce outputs that are fluent, contextually relevant, and plausible. While marking a significant advancement, these L2 systems share a fundamental architectural limitation: their iterative refinement and self-correction processes are governed by the LLM's own internal, opaque, and often unreliable judgment. This leads to the final and most challenging level, L3: Logical Soundness, which requires that an answer is not only factually grounded and coherent but also formally self-consistent and free of logical contradictions. Reaching this pinnacle of reliability remains an open challenge.

To bridge the gap from L2 semantic coherence to L3 logical soundness, we propose LCR-RAG, a reinforcement-learning-driven framework that explicitly optimizes logical consistency in retrieval-augmented generation. The core idea is a Logic-Consistency-driven Reward (LCR), which leverages an external neuro-symbolic verifier (Sawczyn et al., 2025) to transform discrete logical defects—such as contradictions or incomplete inference chains—into structured, high-fidelity reward signals. Concretely, the verifier combines NLI-based factual checking with symbolic reasoning over RDF triples parsed from model outputs, enabling formal detection of logical violations. Guided by this logic-aware reward, a PPO (Proximal Policy Optimization) agent iteratively refines queries by selecting high-level editing actions based on structured verifier feedback, replacing blind self-correction with a targeted, reward-driven optimization process. As a result, logical soundness becomes a direct and verifiable optimization objective rather than an implicit by-product of model self-judgment.

We conduct extensive experiments on four widely used QA benchmarks, including HotpotQA (Yang et al., 2018), ASQA (Stelmakh et al., 2022), MuSiQue (Trivedi et al., 2022), and TriviaQA (Joshi et al., 2017), using five representative LLM backbones: Qwen3-8B (Yang et al., 2025), Llama-3.3-8B (AI@Meta, 2024), DeepSeek-R1-8B (DeepSeek-AI, 2025), GLM-4-9B (GLM et al., 2024), and Ministral-3-8B (AI, 2025). The results demonstrate that LCR-RAG consistently outperforms strong RAG baselines across all datasets and model backbones. In particular, compared to standard RAG pipelines, our method achieves up to 30% relative improvements in answer accuracy and over 10% absolute gains in faithfulness-related metrics, validating the effectiveness of explicitly optimizing logical consistency.Furthermore, ablation studies and hyperparameter analyses confirm that the logic-consistency-driven reward and the PPO-based optimization strategy are the primary contributors to the observed performance gains.

Overall, this work makes three key contributions:

(1) We propose the "RAG Reliability Pyramid" as a conceptual framework for understanding RAG systems and formally define the "logic deficit" that limits the reasoning reliability of current approaches.

(2) We introduce the LCR mechanism, a novel method for converting symbolic logic verification into a direct and optimizable reward signal for reinforcement learning.

(3) We present an end-to-end framework, LCR-RAG, driven by a PPO agent, and provide empirical validation of its state-of-the-art performance, demonstrating a new and effective path toward building more reliable, logically sound, and efficient AI reasoning systems.

2 Preliminaries

We first introduce key concepts and notations for our task.

Retrieval-Augmented Generation (RAG). In a Retrieval-Augmented Generation (RAG) system (Lewis et al., 2020), the task is to improve the answer generation process by grounding the model's output in external knowledge. Given a query q, relevant documents are retrieved from a knowledge corpus $\mathcal{C}$ using a retrieval model, such as Dense Passage Retriever (DPR) (Karpukhin et al., 2020), which encodes both the query and documents into dense vector embeddings. Relevance is measured by cosine similarity between the query's embedding and those of the documents. The top k relevant documents are selected and passed, along with the original query, to a generator model G, typically a large language model (LLM). The generator G then produces an answer y based on both the query and the retrieved documents, ensuring the output is grounded in external knowledge. The whole process can be represented as $y = G(q, D_q)$, where D_q is the set of retrieved documents. This improves both accuracy and contextual grounding.

Proximal Policy Optimization (PPO). In the context of query optimization, PPO (Schulman et al., 2017) is employed as a reinforcement learning technique to improve the quality of the generated queries. The goal is to iteratively refine the query q such that it maximizes the likelihood of obtaining a correct and contextually relevant answer (Wen et al., 2025; Lin et al., 2024). The PPO agent learns a policy π_θ, parameterized by θ, which maps states (representing the query or a modification of it) to actions (such as rephrasing or selecting different documents). This policy is updated to maximize expected cumulative rewards (Jiang et al., 2025), where the reward signal reflects the quality of the answer produced by the query. The update process is done by optimizing a clipped surrogate objective, which helps balance exploration and stability during learning. The update rule ensures that the new policy does not deviate too far from the previous one, preventing instability during training. This method effectively guides the agent to make systematic improvements in query formulation, gradually learning the optimal actions to improve the answer retrieval process.

3 Methodology

To address the persistent challenges of logical inconsistency and inefficient iteration in Retrieval-Augmented Generation (RAG), we propose **LCR-RAG**, a closed-loop framework that integrates dynamic intelligent routing, tiered neuro-symbolic verification, and reinforcement learning-based query optimization. As shown in Figure 2, the system begins with a routing module that classifies queries by complexity, allocating computational resources accordingly. Complex queries are processed through an iterative loop comprising a neuro-symbolic verifier, which assesses answer quality, and a PPO agent that selects high-level query edits based on verifier feedback.

3.1 Dynamic Intelligent Routing

A foundational inefficiency in many advanced RAG systems is their one-size-fits-all approach to processing queries. A simple factual lookup is often subjected to the same computationally intensive reasoning pipeline as a deeply complex query requiring causal inference. To optimize resource allocation and minimize user-perceived latency, LCR-RAG employs a **Dynamic Intelligent Router**. This module serves as a preliminary filter, performing a rapid assessment of query complexity to direct it to the most appropriate and cost-effective processing lane: FAST (for existing cached answers), STANDARD (for single-pass RAG), or DEEP (for the full iterative optimization pipeline).

Feature Extraction. We define a feature extraction function $\Phi(q)$ that maps a raw query q to a numerical feature vector $x_q = \Phi(q)$. The resulting feature vector encapsulates three critical dimensions. First, **Entity Density** ($x_{\textbf{density}}$) measures the ratio of named entities (detected via a fine-tuned spaCy NER model) to the total number of tokens, where higher density often indicates complex entity relationships. Second, **Logical Complexity** ($x_{\textbf{logic}}$) is derived by counting logical keywords (e.g., "and," "or," "why," "compare") and identifying nested clauses using dependency parsing. Finally, **Term Sparsity** ($x_{\textbf{sparsity}}$) is calculated as the average Inverse Document Frequency (IDF) of query terms against a large corpus, where high sparsity implies rare or niche topics requiring multi-hop retrieval.In practice, $\Phi(q)$ includes additional lightweight lexical and syntactic indicators beyond these three key features; we highlight them here for clarity and defer the complete feature list to the Appendix A.

Probabilistic Modeling. The extracted feature vector x_q is passed into a lightweight classifier C_ϕ based on distilled BERT, parameterized by ϕ. The model outputs a softmax distribution over the processing lanes $l \in \{\text{FAST}, \text{STANDARD}, \text{DEEP}\}$:

$$P_\phi(l \mid q) = \text{Softmax}(f_\phi(x_q)) \tag{1}$$

This gives a confidence score for each lane. For example, a query might yield {FAST: 0.1, STANDARD: 0.2, DEEP: 0.7}, indicating high expected complexity.

Decision Rule. The routing lane l^* is selected based on the highest predicted probability:

$$l^* = \arg\max_l P_\phi(l \mid q) \tag{2}$$

To avoid erroneous routing under low confidence, we apply a threshold $\tau = 0.5$. If $P_\phi(l^* \mid q) < \tau$, the query defaults to the STANDARD lane as a safe fallback. This strategy ensures high-efficiency handling for simple queries, while preserving deep reasoning capacity for complex cases.

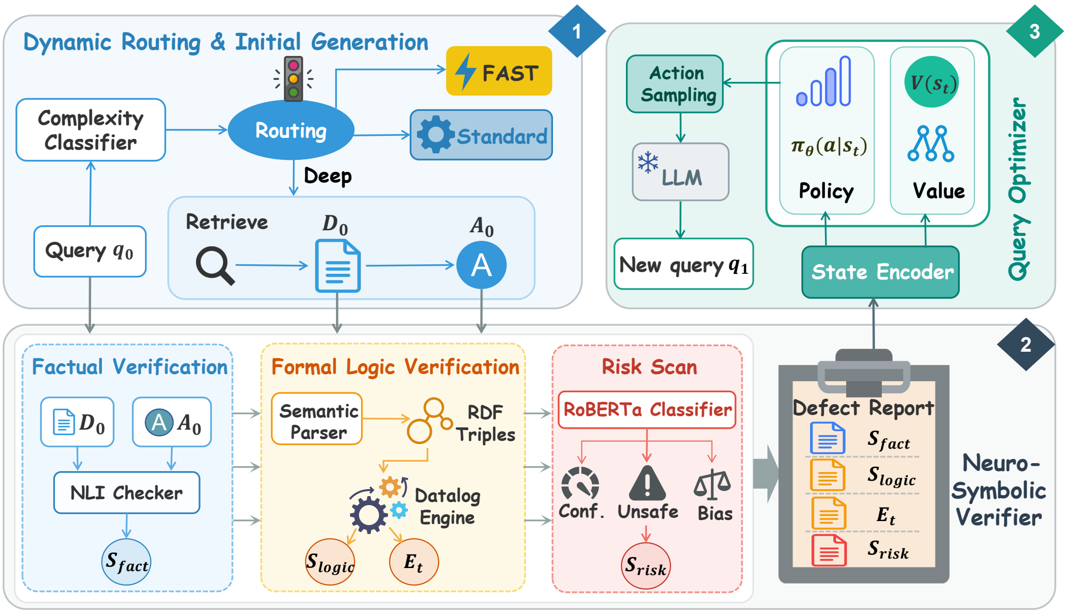

Figure 2: The overall architecture of the LCR-RAG framework. It details the three main stages: (1) Dynamic Routing & Initial Generation, (2) the Neuro-Symbolic Verifier which produces a Defect Report, and (3) the PPO-based Query Optimizer that refines the query.

3.2 Tiered Neuro-symbolic Verifier (NSV)

The **Tiered Neuro-symbolic Verifier** (NSV) is a central component of LCR-RAG's ability to address the logic deficit. Unlike prior RAG systems that rely solely on black-box LLM self-reflection, the NSV explicitly externalizes and formally structures the verification process. It generates a structured, multi-faceted **Defect Report** $\mathcal{L}_t = (S_{\text{fact}}, S_{\text{logic}}, S_{\text{risk}}, E_t)$, offering interpretable diagnostics that guide targeted optimization.

Factual Verification. This stream checks whether the given answer draft A_t is grounded in the retrieved documents D_t. Claims $\{c_1, \ldots, c_n\}$ are extracted from A_t via a dependency-tree-based parser. For each extracted claim c_i, we retrieve a single evidence sentence $e_i \in D_t$, defined as the top-1 sentence ranked by BM25 with respect to c_i. A DeBERTa-v3-large model (He et al., 2021), fine-tuned on NLI data, then estimates entailment probabilities. The final factual grounding score is calculated as:

$$S_{\text{fact}} = \frac{1}{n}\sum_{i=1}^{n} P(y = \text{entailment} \mid c_i, e_i) \tag{3}$$

Formal Logic Verification. This stream detects internal logical inconsistencies in A_t via neuro-symbolic methods. In the **Neuro-Symbolic Translation** stage, a fine-tuned T5-large (Raffel et al., 2020) model f_{T5} is used to map a claim c_i to RDF triples, $f_{\text{T5}}(c_i) \rightarrow \{(s, p, o)\}$ (Ghosh et al., 2025). Following translation, we perform **declarative symbolic reasoning** using a Datalog-inspired rule engine. Extracted triples are added to a temporary knowledge base, over which approximately 150 curated rules $\mathcal{R}$ are evaluated to detect logical inconsistencies and missing inferences. These rules encompass **Logical Axioms** that encode universal relational properties (e.g., transitivity), **Commonsense Constraints** designed to detect contradictions such as simultaneous distinct locations, and **Task-Specific Heuristics** derived from empirical analysis to capture frequent reasoning failures, including incomplete inference chains and circular justifications. The rule engine (Mao et al., 2019) returns a set of discrete logical violation events E_t. Here $E_t \subseteq \mathcal{E}$ is the set of triggered violation types from a predefined event vocabulary $\mathcal{E}$ (e.g., `Contradiction`, `IncompleteChain`, `CircularReasoning`, `EntityMismatch`). The logic score is defined by severity-weighted deductions:

$$S_{\text{logic}} = 1 - \sum_{k \in E_t} w_k, \tag{4}$$

where w_k denotes the predefined severity weight associated with event k.

Risk and Plausibility Scan. A RoBERTa-based classifier scans the answer draft A_t to detect latent risk signals, ranging from speculative phrasing and overconfidence to bias and unsafe content. Fine-tuned on safety-labeled instruction data, the model outputs a scalar probability $S_{\text{risk}} \in [0, 1]$ representing the estimated hazard level. Crucially, we utilize this score as a strict safety gate rather than a shaping reward to prevent the optimization of unsafe trajectories. The final score is then aggregated with factual and logical metrics to constitute the comprehensive Defect Report $\mathcal{L}_t$.

3.3 PPO-based Query Optimizer

To improve upon rigid heuristic exploration, LCR-RAG introduces a PPO-based optimizer that adapts its strategy based on the neuro-symbolic Defect Report ($\mathcal{L}_t$). The framework utilizes a hierarchical design where strategic decision-making is separated from language generation: a lightweight policy network determines the editing operation, which is subsequently carried out by a frozen Qwen3-1.7B model via pre-defined prompt templates (see Appendix J). By isolating the optimization logic from the generative process, this approach balances efficiency with transparent query refinement.

MDP Formulation. The refinement process is modeled as a Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, \mathcal{R}, P, \gamma)$. The **State Space ($\mathcal{S}$)** consists of states s_t, each defined as a fixed-size vector formed by concatenating Sentence-BERT embeddings of the original query q_0, current query q_t, current answer A_t, and the vectorized defect report $\mathcal{L}_t$. The **Action Space ($\mathcal{A}$)** comprises a discrete set of high-level query editing operations, including `LogicalDecompose` to break down complex reasoning chains, `EntityAugment` to inject relevant entities, `RephraseFocus` to shift emphasis or resolve ambiguity, and `NoOp` to terminate refinement. ly, the **Reward Function ($\mathcal{R}$)** assigns a numerical reward at each step that reflects the degree of improvement in factual and logical quality, penalized by a small edit cost:

$$R_t = w_l \cdot \Delta S_{\text{logic}} + w_f \cdot \Delta S_{\text{fact}} - C_{\text{cost}}, \tag{5}$$

where ΔS measures the score change after applying an edit.

Training Procedure. The PPO agent employs a policy network $\pi_\theta(a \mid s)$ and a value network $V_\phi(s)$, both implemented as 3-layer MLPs. Training proceeds in two distinct phases. First, we perform **Bootstrapped Pretraining**, initializing the policy via supervised learning on expert trajectories $\mathcal{D}_{\text{expert}}$ constructed using rule-based mappings from verifier signals to actions (e.g., `IncompleteChain` $\rightarrow$ `LogicalDecompose`). The pretraining objective is standard behavior cloning:

$$\mathcal{L}_{\text{BC}}(\theta) = -\sum_{(s_i, a_i^*)} \log \pi_\theta(a_i^* \mid s_i) \tag{6}$$

Second, we conduct **Online Fine-tuning via PPO**. The policy is optimized using PPO-Clip with the following clipped surrogate loss:

$$\begin{aligned} L_{\text{CLIP}}(\theta) = \mathbb{E}_t\Big[\min\Big(& r_t(\theta)\hat{A}_t, \\ & \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \Big)\Big], \end{aligned} \tag{7}$$

where $r_t(\theta)$ is the importance ratio. The advantage $\hat{A}_t$ is computed using Generalized Advantage Estimation (GAE):

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}, \quad \delta_t = R_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t) \tag{8}$$

Simultaneously, the value network $V_\phi(s)$ is optimized to minimize the temporal difference (TD) error:

$$\mathcal{L}_{\text{value}}(\phi) = \frac{1}{2} \sum_t \left(V_\phi(s_t) - R_t^{\text{target}} \right)^2 \tag{9}$$

This two-phase approach ensures the policy is stably and robustly initialized and capable of generalizing beyond heuristics by leveraging the structured verifier feedback.

4 Experimental Setup

4.1 Datasets.

We evaluate our method on four widely-used benchmarks, each designed to test distinct aspects of complex question answering: HotpotQA (Yang et al., 2018) evaluates multi-hop reasoning through questions that require integrating information across multiple documents; ASQA (Stelmakh et al., 2022) focuses on disambiguation in open-ended question answering; MuSiQue (Trivedi et al., 2022) targets compositional multi-hop reasoning in a curated setting; and TriviaQA (Joshi et al., 2017) assesses open-domain factual QA under noisy retrieval.

4.2 Baselines.

We compare LCR-RAG with a diverse suite of competitive baselines, including: Standard RAG as a vanilla pipeline; ReAct (Yao et al., 2023) and IRCoT (Trivedi et al., 2023) as agent-based and iterative reasoning frameworks; Self-RAG (Asai et al., 2024) as an adaptive and self-corrective retrieval-augmented generation model; SeaKR (Yao et al., 2024) as a selective retrieval strategy; as well as R3-RAG (Li et al., 2025) and RAG-RL (Huang et al., 2025), which incorporate reinforcement learning for reward-driven generation.

4.3 Evaluation Metrics.

We assess model performance using a comprehensive set of metrics that capture both answer quality and factual reliability. Specifically, we use the F1 score on HotpotQA, MuSiQue, and TriviaQA to evaluate lexical overlap with reference answers, and adopt the ROUGE-L metric for ASQA to better reflect performance on ambiguous questions. Additionally, we report Faithfulness (Faith) and Answer Relevance (AnsR) as defined by the RAGAS framework (Es et al., 2024) to measure the degree to which generated answers are grounded in retrieved evidence and semantically aligned with the input query. This multi-faceted evaluation provides a robust assessment of both correctness and factual consistency.

4.4 Implementation Details.

Our LCR-RAG framework employs a frozen Qwen3-8B (Yang et al., 2025) model as the core LLM generator. The PPO agent utilizes a lightweight MLP to learn a query optimization policy from neuro-symbolic feedback. Query edits are executed by a separate, frozen Qwen3-1.7B model, decoupling decision-making from language realization. The verifier's neural components comprise a fine-tuned DeBERTa-v3-large model for factual consistency and a T5-large model for structured logical parsing. All experiments were conducted on NVIDIA A100 GPUs. For full reproducibility, a comprehensive list of implementation details and hyperparameters—including learning rates, PPO clipping thresholds, and LCR reward weights—is provided in the Appendix A.

5 Experimental Results

5.1 Main Results

As shown in Table 1, LCR-RAG achieves consistently higher performance scores across all four QA benchmarks compared to the baselines. Compared to the standard RAG pipeline, it achieves over 30% relative improvement in F1-type metrics, along with more than 10% gains in Faithfulness and Answer Relevance. When evaluated against stronger baselines such as ReAct and IRCoT, LCR-RAG delivers 5–10% higher accuracy, with notable improvements on complex tasks like MuSiQue and open-domain challenges such as TriviaQA. In comparison to reinforcement learning-based models like R3-RAG and RAG-RL, LCR-RAG further improves Faithfulness by approximately 3% on average, highlighting its superior factual alignment. These results demonstrate that LCR-RAG improves answer accuracy while producing semantically relevant and faithfully grounded responses, with robust and empirically consistent gains observed across diverse QA scenarios and multiple LLM backbones, as further reported in Table 7.

5.2 Ablation Study

To rigorously validate our design choices, we conducted three distinct ablation studies on the HotpotQA dataset.

Ablation on Core Architectural Components We first carefully analyzed the impact of removing key modules from our proposed framework. As shown in Figure 3, removing the PPO Optimizer entirely ("w/o PPO Optimizer") results in a sharp decline in performance, with results regressing to the level of a standard single-pass RAG system. This proves that the iterative, learned optimization loop is the fundamental mechanism responsible for effectively correcting errors. Similarly, removing the logic-consistency-driven reward while retaining PPO training ("w/o LCR Reward Signal") also results in a substantial degradation, especially in Faithfulness (a 17-point drop), confirming that our neuro-symbolic reward signal is the crucial ingredient that enables the agent to learn effectively.

Dissecting the Verifier's Contribution To understand which part of our verifier is more critical, we tested ablated versions under a "Contradictory Noise" condition. As shown in Figure 4, removing the Formal Logic check ("w/o Logic Check") causes the Faithfulness score to plummet from a robust 0.88 to a near-failure 0.48. This result indicates that the symbolic reasoning engine is critical to our framework's robustness, confirming that formal logic verification is essential for identifying sophisticated reasoning errors.

	HotpotQA			ASQA			MuSiQue			TriviaQA		
Model	F1	Faith	AnsR	ROUGE-L	Faith	AnsR	F1	Faith	AnsR	F1	Faith	AnsR
Standard RAG	48.5	86.2	85.1	35.8	84.5	83.2	46.2	87.1	89.4	60.2	83.5	84.1
ReAct	56.4	88.2	89.5	40.2	90.1	88.5	50.1	89.5	93.8	67.4	90.4	90.1
IRCoT	53.8	90.5	91.2	36.5	88.4	86.2	46.5	90.8	90.4	68.4	91.8	88.6
Self-RAG	51.2	90.1	89.8	44.1	92.5	91.4	52.3	94.6	94.8	65.1	91.2	89.9
SeaKR	54.5	92.4	90.5	42.8	91.2	89.7	54.8	93.1	95.5	71.8	94.5	92.4
R3-RAG	59.6	91.8	91.2	43.5	92.1	90.8	58.2	93.5	94.2	73.4	93.8	91.5
RAG-RL	55.4	91.2	90.8	41.9	90.8	89.5	59.1	94.8	96.1	69.5	94.6	92.1
LCR-RAG	**63.8**	**94.5**	**93.4**	**48.2**	**94.6**	**92.8**	**62.5**	**96.8**	**97.5**	**78.2**	**97.1**	**94.5**

Table 1: Main results comparing LCR-RAG with state-of-the-art baselines using Qwen3-8B as the backbone generator. Evaluations are conducted across four diverse benchmarks: HotpotQA (multi-hop reasoning), ASQA (long-form ambiguity), MuSiQue (complex connected reasoning), and TriviaQA (fact retrieval). Metrics include F1/ROUGE-L for utility and RAGAS Faithfulness/Answer Relevance for hallucination control. All reported scores are the average of 5 runs.The best-performing model for each metric is highlighted in **bold**.

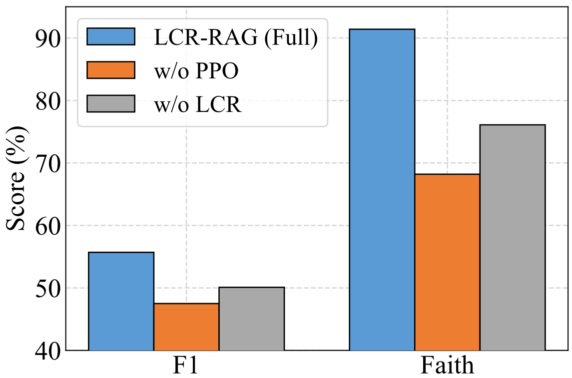

Figure 3: Ablation study on the core architectural components of LCR-RAG.

Validating PPO Agent Generalization Finally, to prove that the PPO agent provides value beyond a fixed set of rules, we created a "long-tail" test set of difficult problems that the rule-based optimizer consistently fails to solve. As shown in Table 2, the rule-based system fails on these problems by definition. The PPO agent, however, solves a significant portion of them, achieving an F1 score of 50.3. This provides powerful evidence that the PPO agent learns to generalize and discover novel solutions for complex problems that lie beyond the scope of predefined rules.

Optimizer Model	F1 Score	Faithfulness
Rule-Based Optimizer	25.8	0.31
PPO Agent (Ours)	**50.3**	**0.85**

Table 2: Performance on "long-tail" problems that the rule-based optimizer fails to solve.

6 Analysis and Discussion

6.1 Hyperparameter Analysis

Effect of Maximum Iteration Budget M We analyze the impact of the maximum iteration budget $M \in \{1, \dots, 10\}$ on both task performance and inference latency. As shown in Figure 5, increasing M yields significant performance gains in the early stages ($M \leq 4$), after which the improvement saturates, indicating a diminishing return. Notably, while the maximum budget increases linearly, the *average* inference latency grows **sub-linearly**. This efficiency gain is attributed to our adaptive early stopping mechanism, which terminates the refinement process for easier queries at early steps, preventing unnecessary computation. Consequently, we select $M = 5$ as the default setting to achieve an optimal balance between quality and efficiency.

Reward Weight Sensitivity We study the sensitivity of LCR-RAG to the reward weight ratio $w_{logic} : w_{fact}$ by analyzing the main task performance (Answer F1) together with the two internal diagnostic signals used in the reward function, namely the logic consistency score S_{logic} and the factual grounding score S_{fact}. As shown in Figure 7, increasing the logic weight ($w_{logic} \gg w_{fact}$) improves S_{logic} but leads to a sharp decline in S_{fact}, indicating reduced factual grounding, while emphasizing factuality ($w_{fact} \gg w_{logic}$) yields the opposite effect. Importantly, as illustrated in Figure 6, the task performance (Answer F1) exhibits a clear **inverted U-shaped trend**, peaking at balanced settings between 2 : 1 and 1 : 2, demonstrating that jointly optimizing logic and factuality is crucial for high-quality reasoning.

6.2 Analysis of Model Scales

To investigate the performance of models with different scales on our method, we conduct comparative experiments using Qwen3 models with 8B and 32B parameters. A more detailed analysis can be found in Appendix H.

7 Related Work

Advanced Iterative RAG Frameworks. To move beyond the limitations of standard "retrieve-and-generate" pipelines, recent research has increasingly focused on dynamic, multi-step frameworks (Borgeaud et al., 2021; Yu et al., 2023). One prominent approach involves agent-based reasoning, where models like ReAct learn to generate discrete thought-action traces (t_i, a_i) to interact with an environment, such as a search engine, in a step-by-step manner. Another significant direction is self-correction, where frameworks like Self-RAG train the LLM to generate special control tokens that trigger an internal critique-and-refinement process based on retrieved evidence. These methods significantly improve semantic coherence and overall task performance, effectively achieving the L2 reliability level of being coherent and plausible. However, their core feedback mechanism relies on the LLM's internal, non-verifiable judgment (Ouyang et al., 2022), which can be unreliable when the model's reasoning is flawed. In contrast, LCR-RAG externalizes this feedback loop, grounding it not in the LLM's opaque self-reflection, but in a verifiable, external neuro-symbolic system that provides explicit logical signals.

Neuro-Symbolic Approaches for LLM Reliability. Neuro-Symbolic (NeSy) (Serafini and d'Avila Garcez, 2016; Manhaeve et al., 2018) methods aim to combine the pattern-recognition strengths of neural networks with the formal rigor of symbolic systems. In the context of LLMs, this often involves two distinct stages. First, a neural component is used to interface with natural language; for example, a T5 model can be fine-tuned as a *learned* semantic parser f_{parser} to convert a natural language claim c into a structured logical representation like an RDF triple, such that $f_{\text{parser}} : c \to \{(s, p, o)\}$. Second, a symbolic engine (e.g., a Datalog reasoner) applies a *predefined* formal rule base $\mathcal{R}$ to this structured data to perform deductions or check for inconsistencies. This approach provides a high-fidelity method for *explicitly* verifying logical properties (Glanois et al., 2022; Liu et al., 2025). LCR-RAG uniquely synthesizes these NeSy principles into its verifier module; it uses an NLI model for factual grounding and a T5–Datalog pipeline for logical validation, then innovatively translates the structured output of this entire verifier into a quantitative reward signal to guide and shape the behavior of our reinforcement learning agent.

Reinforcement Learning in RAG. Several recent studies now enhance Retrieval-Augmented Generation (RAG) through reinforcement and curriculum learning. RAG-RL applies reinforcement learning for query optimization, aligning with our use of PPO, though it lacks verifiable feedback from logical consistency checks. CL-RAG introduces curriculum learning to train query policies progressively; our framework instead leverages direct, logic-aware reward signals for adaptive policy updates. LevelRAG focuses on logical decomposition in multi-hop QA, which conceptually overlaps with our logical actions but omits policy learning from structured feedback. Finally, ReaRAG improves factuality via iterative retrieval and heuristic rewriting, whereas our PPO agent dynamically adjusts queries using a neuro-symbolic verifier. In contrast to these methods, LCR-RAG uniquely transforms formal logic violations into direct reward signals, enabling verifiable and self-correcting learning beyond these static or heuristic-driven designs.

8 Conclusion

In this work, we propose LCR-RAG, a novel framework designed to address the critical challenges of logical consistency and inefficient iteration in complex reasoning tasks for Retrieval-Augmented Generation (RAG) systems. By integrating an external neuro-symbolic verifier with reinforcement learning, LCR-RAG introduces a Logic-Consistency Reward (LCR) mechanism, which translates logical and factual errors into structured feedback, enabling precise query refinement. Through a combination of rule-based directed optimization and a PPO-driven exploratory optimizer, our approach iteratively improves reasoning quality, ensuring that logical soundness becomes a direct optimization target. Our experimental results on benchmarks such as HotpotQA and ASQA demonstrate that LCR-RAG significantly outperforms state-of-the-art baselines, including RAG-RL and SeaKR, across multiple evaluation metrics. Furthermore, our ablation studies highlight the importance of each component in achieving robust and accurate reasoning. LCR-RAG thus represents a promising approach toward developing more reliable, interpretable, and efficient AI systems capable of tackling intricate multi-step reasoning problems.

Limitations

Despite our efforts to thoroughly evaluate LCR-RAG, several limitations remain. First, while the neuro-symbolic verifier provides explicit logic-aware feedback, the current symbolic rule set is manually designed and fixed, which may limit adaptability to new domains. Extending the framework with automatically induced or adaptive rules is a promising direction for future work. Second, although LCR-RAG improves logical consistency through iterative query optimization, it does not yet perform deeper reasoning verification and correction beyond the current optimization loop, which we leave for future exploration.

Acknowledgments

The research in this article is supported by the National Natural Science Foundation of China (U22B2059, 62276083), Key Research and Development Program of Heilongjiang Province (2022ZX01A28). Besides, We would like to express our gratitude to Du Xiaoman Technology for their support of our research paper.

References

Mistral AI. 2025. Introducing mistral 3. https://mistral.ai/news/mistral-3.

AI@Meta. 2024. Llama 3 model card.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, T. W. Hennigan, Saffron Huang, Lorenzo Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2021. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*.

Fengxiang Cheng, Haoxuan Li, Fenrong Liu, Robert van Rooij, Kun Zhang, and Zhouchen Lin. 2025. Empowering llms with logical reasoning: A comprehensive survey. *Preprint*, arXiv:2502.15652.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAs: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023a. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *ArXiv*, abs/2312.10997.

Bishwamittra Ghosh, Sarah Hasan, Naheed Anjum Arafat, and Arijit Khan. 2025. Logical consistency of large language models in fact-checking. In *The Thirteenth International Conference on Learning Representations*.

Claire Glanois, Zhaohui Jiang, Xuening Feng, Paul Weng, Matthieu Zimmer, Dong Li, Wulong Liu, and Jianye Hao. 2022. Neuro-symbolic hierarchical rule induction. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 7583–7615. PMLR.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, and 37 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Jerry Huang, Siddarth Madala, Risham Sidhu, Cheng Niu, Hao Peng, Julia Hockenmaier, and Tong Zhang. 2025. Rag-rl: Advancing retrieval-augmented generation via rl and curriculum learning. *Preprint*, arXiv:2503.12759.

Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2025. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning. *Preprint*, arXiv:2503.00223.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Yuan Li, Qi Luo, Xiaonan Li, Bufan Li, Qinyuan Cheng, Bo Wang, Yining Zheng, Yuxin Wang, Zhangyue Yin, and Xipeng Qiu. 2025. R3-rag: Learning step-by-step reasoning and retrieval for llms via reinforcement learning. *Preprint*, arXiv:2505.23794.

Sheng-Chieh Lin, Luyu Gao, Barlas Oguz, Wenhan Xiong, Jimmy Lin, Wen-tau Yih, and Xilun Chen. 2024. Flame: Factuality-aware alignment for large language models. *Advances in Neural Information Processing Systems*, 37:115588–115614.

Yinhong Liu, Zhijiang Guo, Tianya Liang, Ehsan Shareghi, Ivan Vulić, and Nigel Collier. 2025. Aligning with logic: Measuring, evaluating and improving logical preference consistency in large language models. *Preprint*, arXiv:2410.02205.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. 2018. Deepproblog: neural probabilistic logic programming. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 3753–3763, Red Hook, NY, USA. Curran Associates Inc.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. 2019. The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision. In *International Conference on Learning Representations*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

Albert Sawczyn, Jakub Binkowski, Denis Janiak, Bogdan Gabrys, and Tomasz Kajdanowicz. 2025. Fact-selfcheck: Fact-level black-box hallucination detection for llms. *Preprint*, arXiv:2503.17229.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Luciano Serafini and Artur d'Avila Garcez. 2016. Logic tensor networks: Deep learning and logical reasoning from data and knowledge. *Preprint*, arXiv:1606.04422.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Harsh Trivedi, Niranjan 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, Niranjan 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, Toronto, Canada. Association for Computational Linguistics.

Xueru Wen, Jie Lou, Xinyu Lu, Ji Yuqiu, Xinyan Guan, Yaojie Lu, Hongyu Lin, Ben He, Xianpei Han, Debing Zhang, and Le Sun. 2025. On-policy self-alignment with fine-grained knowledge feedback for hallucination mitigation. *Preprint*, arXiv:2406.12221.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Zijun Yao, Weijian Qi, Liangming Pan, Shulin Cao, Linmei Hu, Weichuan Liu, Lei Hou, and Juanzi Li. 2024. Seakr: Self-aware knowledge retrieval for adaptive retrieval augmented generation. *Preprint*, arXiv:2406.19215.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023. Generate rather than retrieve: Large language models are strong context generators. In *International Conference for Learning Representation (ICLR)*.

A Implementation Details and Hyperparameters

System Overview and Infrastructure. Our implementation is built upon the Hugging Face `transformers` and `trl` libraries. All experiments are conducted on NVIDIA A100-80G GPUs. For the retrieval component, we employ a recursive character splitting strategy (chunk size=200, overlap=20) and index the corpus using a FAISS vector store. During inference, we retrieve the top-k ($k = 5$) documents based on cosine similarity.

Model Architectures. The core components of LCR-RAG are instantiated as follows:

- **Generator:** We use a frozen `Qwen3-8B` as the backbone for generation.
- **Policy Network:** The PPO value head is a 3-layer MLP with hidden dimensions $[256, 128, 64]$ and ReLU activations.
- **Dynamic Router:** A distilled `BERT-base-uncased` model serves as the router, taking 12 linguistic features concatenated with the query embedding.
- **Neuro-Symbolic Verifier (NSV):** We utilize `DeBERTa-v3-large` for Natural Language Inference (NLI) to verify claim entailment. The semantic parser is a `T5-large` model fine-tuned to translate natural language claims into RDF triples.

Component	Parameter	Value
PPO Optimizer	Discount Factor (γ)	0.99
	Learning Rate (α)	1×10^{-4}
	Batch Size	128
	PPO Clip Epsilon (ϵ)	0.2
	GAE Lambda (λ)	0.95
Reward Function	Logic Weight (w_l)	0.4
	Factual Weight (w_f)	0.6
	Step Cost (C_{cost})	0.01
Dynamic Router	Architecture	Distilled BERT
	Learning Rate	2×10^{-5}
	Confidence Threshold (τ)	0.5
NSV Modules	NLI Threshold (τ_{entail})	0.9
	Parser Model	T5-large
	Fine-tuning Epochs	3

Table 3: Detailed hyperparameters for the LCR-RAG framework, including PPO optimization settings, reward coefficients, and threshold configurations for the auxiliary modules.

Feature Extraction for Routing. To efficiently route queries to the appropriate processing lane (FAST, STANDARD, or DEEP) without heavy computation, we extract a feature vector $\Phi(q) \in \mathbb{R}^{12}$. As detailed in Table 4, these features are categorized into three groups: *Core Complexity* (indicating reasoning depth), *Structural & Syntactic* (quantifying linguistic intricacy via dependency parsing), and *Semantic Intent* (classifying specific reasoning types like temporal or comparative logic).

Training and Hyperparameters. The Dynamic Router is trained using standard cross-entropy loss. The NSV components (T5 parser) are fine-tuned for 3 epochs on a mixture of WebNLG and synthetic data. The PPO agent is trained using the AdamW optimizer. We report the detailed hyperparameters used for the PPO optimizer, reward function configuration, and auxiliary models in Table 3.

B The Datalog Rule Base

Our Datalog knowledge base consists of approximately 150 manually curated rules designed to capture high-impact inconsistency patterns.

Rule Categories.

Category	Feature Name	Definition & Intuition
Core Complexity	Entity Density (x_{den})	Ratio of Named Entities to total tokens. High density implies multi-entity constraints that may require complex verify-steps.
	Logical Keywords	Normalized frequency of connectives (e.g., “if”, “unless”, “therefore”); signals conditional logic chains.
	Term Sparsity (x_{spa})	Average IDF of query terms. High values indicate niche topics necessitating multi-hop retrieval.
Structural & Syntactic	Syntactic Tree Depth	Maximum depth of the dependency parse tree; a strong proxy for nested query structure.
	Clause Count	Number of subordinate clauses (e.g., SBAR tags), indicating embedded sub-questions.
	Query Length	Total token count; serves as a baseline metric for estimating processing cost.
	Verb Count	Count of unique verbs; correlates with the number of required action steps.
	Modifier Count	Count of adjectives/adverbs that impose strict constraints on the answer.
Semantic Intent	Comparative Ind.	Binary presence of comparison terms (e.g., “better”, “versus”, “difference”).
	Temporal Ind.	Count of time-related terms (e.g., “year”, “during”); signals temporal reasoning requirements.
	Reasoning “Wh-”	Binary indicator for reasoning-heavy interrogatives (“Why”, “How”) versus fact-seeking ones.
	Numerical Presence	Count of digits or number words; suggests a need for precise numerical verification.

Table 4: The 12-dimensional linguistic feature set used by the Dynamic Router. By leveraging the broader page width, we detail the intuition behind each feature used to classify query complexity.

- **Foundational Axioms (30 rules):** logical properties including transitivity ($A \to B \wedge B \to C \implies A \to C$), symmetry, and reflexivity.
- **Commonsense Constraints (50 rules):** detection of physical impossibilities or mutually exclusive attributes (e.g., an entity cannot be born in two different countries).
- **Task-Specific Heuristics (70 rules):** patterns specific to QA, such as circular reasoning chains or ungrounded speculative jumps.

Penalties. Table 5 shows the penalties applied when rules are violated.

Logical Event	Penalty Weight
Contradiction	0.5
IncompleteChain	0.3
CircularReasoning	0.2
EntityMismatch	0.2

Table 5: Penalty weights for detected logical defects.

C Router Performance

Table 6 confirms the efficiency of our Dynamic Intelligent Router. By effectively filtering simple queries, the computationally expensive DEEP path is triggered for less than 45% of queries on average. The low Expected Calibration Error (ECE < 0.07) indicates reliable confidence estimation.

Metric	HotpotQA	ASQA	CoQA
DEEP Ratio (%)	31.0	42.0	28.0
Accuracy (%)	94.2	91.5	95.1
ECE	0.045	0.061	0.039

Table 6: Dynamic Router performance statistics.

D Verifier Ablation

We further dissect the contribution of the Verifier's components. Figure 4 illustrates that under "Contradiction Noise" conditions, removing the Formal Logic check (“w/o Logic Check”) causes the Faithfulness score to plummet from 0.88 to 0.58. This indicates that while the NLI model handles basic fact-checking, the symbolic reasoning engine is critical for identifying sophisticated logical inconsistencies.

E Effect of Maximum Iteration Budget M

We analyze the impact of the maximum iteration budget $M \in \{1, \ldots, 10\}$. As shown in Figure 5, increasing M yields significant performance gains in early stages ($M \leq 4$). However, the improvement saturates beyond $M = 5$, while inference latency

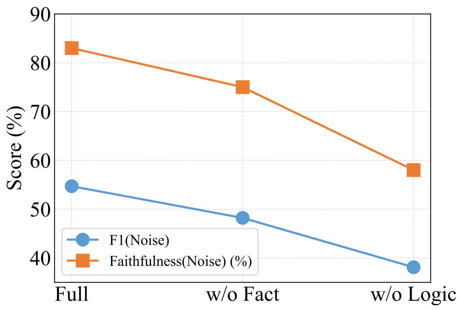

Figure 4: Ablation study dissecting the contribution of the Verifier's components. The symbolic logic check is crucial for robustness against contradictory noise.

continues to grow. We therefore select $M = 5$ as the default setting to achieve an optimal balance between quality and efficiency.

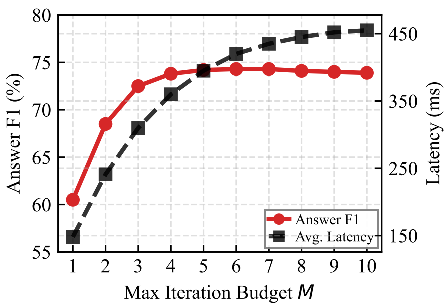

Figure 5: Impact of the maximum iteration budget M on task performance (F1) and inference latency.

F Reward Weight Sensitivity

We investigate the sensitivity of the optimization to the reward weight ratio $w_{logic} : w_{fact}$. Figure 7 reveals a clear trade-off mechanism. A dominant factual weight ($w_f \gg w_l$) degrades logical coherence, while excessive logic weight leads to hallucination. The task performance (Answer F1) exhibits an inverted U-shaped trend, peaking at balanced configurations.

G Cross-Model Generalization Analysis

To evaluate the robustness and model-agnostic nature of LCR-RAG, we conduct cross-model generalization experiments using five representative LLM backbones: Qwen3-8B (Yang et al., 2025), Llama-3.3-8B (AI@Meta, 2024), DeepSeek-R1-8B (DeepSeek-AI, 2025), GLM-4-9B (GLM et al., 2024) and Ministral-3-8B (AI, 2025).

We compare LCR-RAG against a diverse set of strong RAG baselines, including agent-based reasoning methods (ReAct, IRCoT), self-reflective and adaptive retrieval models (Self-RAG, SeaKR), as well as reinforcement-learning-based approaches (R3-RAG and RAG-RL).

Table 7 reports the results across three challenging multi-hop QA benchmarks. We observe that LCR-RAG consistently achieves the best performance across all datasets and model backbones. Notably, the relative improvements remain substantial even for smaller models such as Ministral-3-8B and reasoning-specialized models like DeepSeek-R1-8B. These results indicate that the proposed neuro-symbolic logic-consistency reward is largely model-agnostic and can reliably enhance logical soundness and factual alignment across heterogeneous LLM architectures.

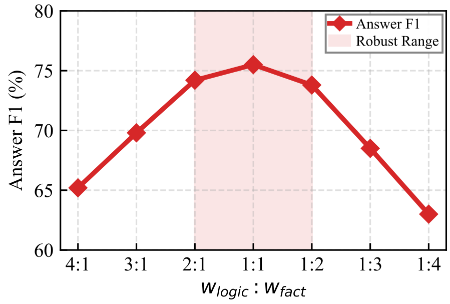

Figure 6: Effect of the reward weight ratio $w_{logic} : w_{fact}$ on Answer F1.

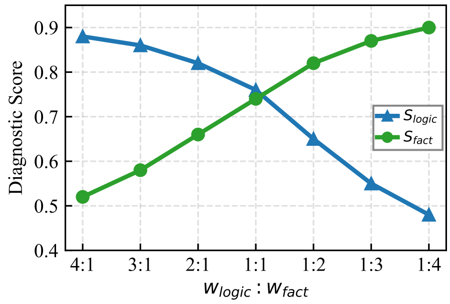

Figure 7: Trade-off between S_{logic} and S_{fact} under different reward weight ratios.

H Parameter Scaling Analysis

We investigate the performance of models with different parameter scales using Qwen3 (8B vs. 32B). As shown in Table 8, increasing model size brings consistent improvements across all metrics. Importantly, the relative improvement (Δ Improve) remains significant even for the larger model, suggesting that our method does not suffer from diminishing returns and scales effectively with model capacity.

I Qualitative Case Studies

In this section, we provide a detailed qualitative analysis to illustrate how LCR-RAG handles complex multi-hop queries that typically confound standard RAG baselines.

Table 9 presents a comparison between Self-RAG and LCR-RAG on a query requiring compositional reasoning about film directors and actors. As shown, the baseline **Self-RAG** retrieves relevant entities (Barry Levinson, Top Gun) but fails to construct a logically valid chain, hallucinating a connection between the director and the actor. The internal self-reflection mechanism fails to detect this factual inconsistency.

In contrast, **LCR-RAG** demonstrates a successful correction loop:

1. **Detection:** The Neuro-Symbolic Verifier (NSV) identifies a specific `Contradiction` in the initial draft (A_0) by verifying the director-film relations against retrieved evidence.

2. **Strategy:** The PPO Agent receives this negative logic reward and selects the `RephraseFocus` action to clarify the director-actor intersection.

3. **Correction:** The rewritten query (q_1) leads to precise retrieval, resulting in a factually and logically sound final answer (A_1).

J Prompt Templates

The prompt templates are presented in Tables 10 and 11.

LLM	Methods	HotpotQA			MuSiQue			TriviaQA		
		F1	Faith	AnsR	F1	Faith	AnsR	F1	Faith	AnsR
Qwen3-8B	ReAct	56.4	88.2	89.5	50.1	89.5	93.8	62.5	90.4	90.1
	IRCoT	53.8	90.5	91.2	46.5	90.8	90.4	68.4	91.8	88.6
	Self-RAG	51.2	90.1	89.8	52.3	94.6	94.8	65.1	91.2	89.9
	SeaKR	54.5	92.4	90.5	54.8	93.1	95.5	71.8	94.5	92.4
	R3-RAG	59.6	91.8	91.2	58.2	93.5	94.2	73.4	93.8	91.5
	RAG-RL	55.4	91.2	90.8	59.1	94.8	96.1	69.5	94.6	92.1
	LCR-RAG	63.8	94.5	93.4	62.5	96.8	97.5	78.2	97.1	94.5
Llama-3.3-8B	ReAct	53.2	86.5	85.4	48.5	87.2	92.8	58.4	88.5	87.2
	IRCoT	50.4	89.2	87.6	44.2	89.1	89.9	64.2	90.1	86.5
	Self-RAG	48.6	88.8	87.2	49.8	93.5	94.5	60.5	89.8	88.4
	SeaKR	51.5	90.5	88.1	51.6	92.4	95.1	67.8	93.4	90.2
	R3-RAG	56.2	90.4	89.5	55.4	92.8	95.4	69.1	93.8	90.8
	RAG-RL	52.8	90.1	88.9	56.8	94.3	95.9	64.5	94.2	90.1
	LCR-RAG	59.5	92.8	90.6	60.2	95.8	96.5	73.4	96.8	92.3
DeepSeek-R1-8B	ReAct	58.1	89.1	87.5	52.4	89.8	93.6	60.5	90.2	88.9
	IRCoT	55.4	91.8	89.4	48.6	91.2	91.5	69.2	92.4	87.8
	Self-RAG	54.8	91.5	90.2	55.1	95.2	95.6	66.8	92.5	90.5
	SeaKR	57.2	93.1	90.8	57.5	94.1	96.2	72.5	95.1	92.1
	R3-RAG	61.5	93.4	91.6	60.2	94.8	96.5	74.8	95.5	92.8
	RAG-RL	58.6	92.8	91.1	61.4	95.9	97.1	70.2	95.8	92.4
	LCR-RAG	**65.4**	**95.2**	**93.8**	**64.8**	**97.1**	**97.6**	**79.1**	**97.5**	**94.8**
GLM-4-9B	ReAct	54.8	87.2	86.8	49.4	88.1	93.2	59.8	89.2	88.5
	IRCoT	51.5	90.1	88.5	45.1	89.8	90.2	65.6	90.8	87.4
	Self-RAG	49.8	89.4	88.1	50.6	94.1	94.9	62.4	90.5	89.1
	SeaKR	52.9	91.2	89.4	52.8	92.9	95.3	69.1	94.1	91.5
	R3-RAG	57.6	91.5	90.2	56.5	93.2	95.7	71.2	94.4	91.8
	RAG-RL	54.2	90.8	89.6	57.9	94.6	96.2	66.8	94.8	91.2
	LCR-RAG	61.2	93.4	91.5	61.4	96.2	96.9	75.6	97.0	93.2
Ministral-3-8B	ReAct	50.5	84.8	82.5	47.2	86.5	91.8	54.2	86.9	85.6
	IRCoT	47.1	88.2	85.8	42.5	88.6	89.1	60.8	89.2	84.5
	Self-RAG	44.8	87.9	86.2	47.1	93.2	93.8	57.5	88.8	87.2
	SeaKR	46.2	90.1	87.5	49.5	91.4	94.5	63.4	92.5	89.5
	R3-RAG	52.5	90.5	88.4	52.8	92.5	95.2	65.1	92.8	90.1
	RAG-RL	48.9	89.8	87.6	53.4	94.1	95.5	61.2	93.9	89.6
	LCR-RAG	57.4	91.8	89.5	59.1	94.8	96.1	70.5	95.6	90.8

Table 7: Cross-model generalization results across five LLM backbones. LCR-RAG consistently outperforms agent-based, self-reflective, and reinforcement-learning-based RAG baselines. Best results within each LLM block are underlined, and best overall results are highlighted in **bold**.

Model	HotpotQA			MuSiQue			TriviaQA		
	F1	Faith	AnsR	F1	Faith	AnsR	F1	Faith	AnsR
LCR-RAG (8B)	63.8	94.5	93.4	62.5	96.8	97.5	78.2	97.1	94.5
LCR-RAG (32B)	66.5	95.6	94.2	65.1	97.5	98.1	81.4	97.8	95.3
Δ *Improve*	+4.23%	+1.16%	+0.86%	+4.16%	+0.72%	+0.62%	+4.09%	+0.72%	+0.85%

Table 8: Performance scaling analysis of LCR-RAG with different parameter scales. The 8B model corresponds to the main results reported in Table 1. Larger models consistently yield better reasoning and retrieval utilization.

Query: Which film directed by a filmmaker who also directed a movie starring Tom Cruise, won the Best Picture Oscar?

Model	Step	Trace / Output
Self-RAG	*Generation*	The film *Rain Man*, directed by Barry Levinson, won the Best Picture Oscar. Barry Levinson also directed *Top Gun*, which starred Tom Cruise.
	Critique	**[Failed]** This answer is marked as fully supported, despite the factual error (Levinson did not direct *Top Gun*).
LCR-RAG	**Draft** (A_0)	The movie is *The Departed*. It was directed by Martin Scorsese, who also directed *Minority Report* starring Tom Cruise.
	Verifier (NSV)	**[Defect Report]**
		• **Event**: `Contradiction`
		• **Reason**: Knowledge Base states *Minority Report* directed by Steven Spielberg, not Scorsese.
		• **Scores**: $S_{logic} = 0.5$, $S_{fact} = 0.5$
	PPO Agent	$\rightarrow$ **Action**: `RephraseFocus` (Target: Director intersection)
	New Query (q_1)	"Which director of a Best Picture winning film also directed a film with Tom Cruise?"
	Answer (A_1)	The film that won the Best Picture Oscar is *Rain Man*, directed by Barry Levinson. Levinson also directed *Rain Man*, which starred Tom Cruise. **[Verified]**

Table 9: Step-by-step execution trace on a complex HotpotQA query. While Self-RAG hallucinates a relationship to force an answer, LCR-RAG's neuro-symbolic verifier explicitly catches the director mismatch in the first draft. This triggers the PPO agent to refine the query, leading to a logically consistent final answer.

System Prompt: High-level Action Selection
You are an expert query optimizer. Your goal is to refine the user's query to resolve logical inconsistencies or factual gaps identified in the current answer draft. **Original Query**: $\{original\_query\}$ **Current Answer Draft**: $\{current\_answer\_draft\}$ **Verifier Feedback**: - Factual Score: $\{factual\_score\}$ - Logical Score: $\{logical\_score\}$ - Detected Logic Events: $\{logic\_events\_summary\}$ Based on the feedback, choose the best high-level optimization action: 1. LogicalDecompose (Break down complex reasoning) 2. EntityAugment (Inject missing entities) 3. RephraseFocus (Adjust query focus) 4. NoOp (Current answer is sufficient) **Action**:

Table 10: The prompt template used by the PPO agent to select the optimization strategy based on verifier feedback.

Execution Prompt: Entity Augment
The following answer contains logical gaps due to missing entities identified by the external knowledge base. **Context**: $\{retrieved\_docs\}$ **Original Answer**: $\{current\_answer\_draft\}$ **Missing Entities**: $\{missing\_entities\_from\_verifier\}$ **Instruction**: Rewrite the Original Answer to include the Missing Entities coherently. Ensure that the logical flow is preserved and that the new entities are grounded in the Context. **Refined Answer**:

Table 11: An example execution prompt for the `EntityAugment` action. Similar templates are used for `LogicalDecompose` and `RephraseFocus`.