RAG-on-a-Diet: A Reinforcement Learning-Based Dynamic Resource Optimization Framework for RAG

Hongwen Ding[1] **and Yizheng Zhao**[2]*
[1]Shenzhen College of International Education
[2]State Key Laboratory of Novel Software Technology, Nanjing University

Abstract

Retrieval-Augmented Generation (RAG) has become the backbone of knowledge-intensive multi-hop question answering, yet routing every sub-query through a frontier model turns every hop into a cost multiplier and makes real-world deployment prohibitively expensive. Existing remedies either fix the retrieval schedule, route once at the query level, or lack a principled stopping rule, leaving a critical gap: *no framework adapts, hop by hop, to how a trajectory actually unfolds*. We introduce **RAG-on-a-Diet**, a lightweight reinforcement-learning agent that treats each reasoning hop as an independent decision and selects the smallest model (Qwen3-4B, Qwen3-30B, or DS-R1-671B) sufficient for it, guided by entity- and confidence-aware features. Trained via behavior cloning followed by PPO under a five-component cost-aware reward (final, cumulative, step-wise, cost, balance) and coupled with an explicit two-tier termination policy (5-hop cap plus a $\tau = 0.3$ confidence gate), the agent carves a Pareto-optimal efficiency frontier. On HotpotQA it cuts Monetary Inference Cost by 60.07% against IRCoT with only a 3.7% F1 drop; it matches Adaptive-RAG's F1 at 37.30% lower cost; and it attains up to 2.33× higher Quality-per-Monetary-Cost. Consistent gains on MuSiQue, 2WikiMultiHopQA, CRAG, and Bamboogle confirm strong out-of-distribution robustness, setting a new paradigm for fine-grained resource control in multi-hop RAG.

1 Introduction

Large language models (LLMs) memorize a remarkable amount of world knowledge yet hallucinate on facts beyond their parameters (Huang et al., 2025). Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Guu et al., 2020) grounds generation in external corpora and has become the de facto recipe for knowledge-intensive tasks, with multi-hop question answering (Yang et al., 2018) as its canonical testbed. As RAG moves from demo to deployment, a new bottleneck dominates: *computational expense*. Routing every sub-query through a frontier model at every hop is financially and environmentally unsustainable. We argue that today's multi-hop RAG suffers from three intertwined contradictions that together point to a single missing ingredient.

(1) Accuracy-cost imbalance at the wrong granularity. Iterative pipelines such as IRCoT (Trivedi et al., 2023), DualRAG (Cheng et al., 2025), and DeepRAG (Guan et al., 2025) invoke the same large model at every hop, inflating the cost of easy queries. Query-level routers such as Adaptive-RAG (Jeong et al., 2024) and Router-R1 (Zhang et al., 2025) commit to a single decision at trajectory start and cannot react to how sub-task difficulty evolves hop by hop.

(2) Routing signals decoupled from reasoning semantics. FrugalRAG (Java et al., 2025) and TreeHop (Li et al., 2025) tune only retrieval. Model-level routers like FrugalGPT (Chen et al., 2024), Hybrid LLM (Ding et al., 2024), and BEST-Route (Ding et al., 2025) allocate capacity by generic query difficulty, blind to the entity importance, relation salience, and intermediate confidence that actually govern multi-hop reasoning. Efficiency-focused designs such as Think-Straight (Bang et al., 2025) cache prefixes yet do not schedule model capacity.

(3) Missing explicit termination under uncertainty. AT-RAG (Rezaei et al., 2024), Replace-Don't-Expand (Lahmy and Yozevitch, 2025), and Re3MHQA (Cao et al., 2025) recognize that futile iterations waste compute, yet none combines a hard hop cap with a calibrated confidence gate; long chains silently accumulate errors and burn tokens past the point of diminishing return.

*Corresponding Author: zhaoyz@nju.edu.cn

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 33883–33904

These gaps converge on one diagnosis: what is missing is *intra-trajectory, hop-wise adaptation* that fuses entity-level semantics with real-time reasoning confidence. We propose **RAG-on-a-Diet**, a lightweight reinforcement-learning agent that treats every hop as an independent decision point and selects the most economical model (Qwen3-4B, Qwen3-30B, or DS-R1-671B) sufficient for that hop, guided by four static and three temporal features. Our contributions are:

- We introduce the first **hop-level dynamic routing** framework for multi-hop RAG, shifting adaptation from one decision per trajectory to one decision per reasoning step.
- We design a five-component **cost-aware reward** (final, cumulative, step-wise, cost, balance) that resolves the sparse-reward and credit-assignment difficulties of long trajectories, trained via behavior cloning followed by PPO in the spirit of recent imitation-then-RL schemes (Macuglia et al., 2025) and process-reward supervision (Wang et al., 2024; Lightman et al., 2024).
- We couple this with an **explicit termination** policy (5-hop hard cap plus confidence threshold $\tau = 0.3$). On HotpotQA, RAG-on-a-Diet cuts cost by **60.07%** against IRCoT with only a **3.7% F1 drop** (0.7671→0.7387), and matches Adaptive-RAG's F1 while **reducing cost by 37.30%**; consistent gains on MuSiQue, 2WikiMultiHopQA, CRAG, and Bamboogle confirm out-of-distribution robustness.

To the best of our knowledge, RAG-on-a-Diet is the first work to perform hop-wise control *inside* a multi-hop trajectory, establishing a new paradigm for fine-grained resource optimization in RAG.

2 Related Work

2.1 From One-Shot to Adaptive Multi-Hop RAG

Classical RAG (Lewis et al., 2020; Guu et al., 2020) issues a single retrieve-then-read pass, insufficient when evidence spans documents. Chain-of-thought (Wei et al., 2022) and ReAct (Yao et al., 2023) enabled interleaved reasoning and action; IRCoT (Trivedi et al., 2023) instantiated this for multi-hop QA with a *fixed* retrieval schedule. Reactive variants then gate retrieval on uncertainty: FLARE (Jiang et al., 2023) triggers on low-confidence tokens; Self-RAG (Asai et al., 2024) and CRAG (Yan et al., 2024) use reflection or evaluator tokens; Adaptive-RAG (Jeong et al., 2024) classifies complexity up-front. All still act at query granularity with a fixed generator.

2.2 Entity- and Decomposition-Driven Reasoning

A parallel thread decomposes hard questions into simpler sub-queries (Decomposed Prompting (Khot et al., 2023), Least-to-Most (Zhou et al., 2023), Self-Ask (Press et al., 2023), which also provides the Bamboogle benchmark). Entity-centric methods refine what to retrieve: DualRAG (Cheng et al., 2025) rewrites queries via key entities, FE2H (Li et al., 2023) filters evidence in two stages, and AT-RAG (Rezaei et al., 2024) prunes via topic filters. These signals, however, are spent on retrieval selection, never on model invocation.

2.3 Cost-Aware Routing, RL for Search, and Termination

Cost-aware LLM routing evolved from cascades and binary classifiers (Chen et al., 2024; Ding et al., 2024) to compute-aware test-time schedulers (Ding et al., 2025), but all make one routing choice per query. A newer wave trains LLMs with RL (Schulman et al., 2017; Guo et al., 2025) to interleave reasoning with search: Router-R1 (Zhang et al., 2025), R1-Searcher (Song et al., 2025a), Smart-Searcher (Song et al., 2025b), and DeepResearcher (Zheng et al., 2025) optimize *when* or *what* to retrieve, while FrugalRAG (Java et al., 2025), TreeHop (Li et al., 2025), and GlobalRAG (Luo et al., 2025a) target retrieval efficiency. Termination is either ignored or left to a single soft signal (Bang et al., 2025; Cao et al., 2025), with no hard hop cap.

Across all three threads, no prior work performs *intra-trajectory, hop-wise* adaptation that jointly uses entity-level semantics and real-time confidence, nor couples it with an explicit two-tier stopping policy. RAG-on-a-Diet closes this gap.

3 Method

3.1 Method Motivation and Core Idea

To address complex queries, multi-turn retrieval strategies have been proposed. For instance, IRCoT (Trivedi et al., 2023) constructs reasoning paths via chain-of-thought and iterative retrieval, while Adaptive-RAG (Jeong et al., 2024) uses complexity classification to dynamically choose the

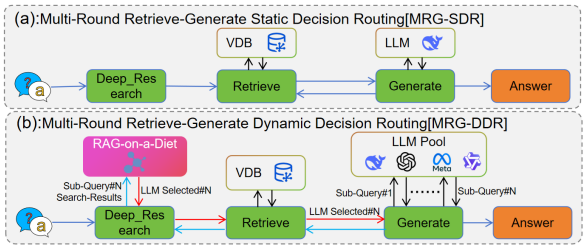

Figure 1: Evolution of Multi-Round Retrieval-Generation Architectures: From Static to Dynamic Routing.
(a) Static Routing: A single fixed model processes all sub-queries, ignoring their varying complexity.
(b) Dynamic Hop-wise Routing (Ours): Each sub-query in the reasoning chain is matched to an optimal model based on its instantaneous complexity and confidence, enabling fine-grained resource control.

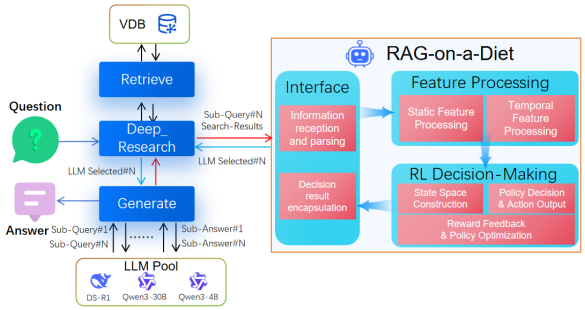

Figure 2: RAG-on-a-Diet: Architecture and Dynamic Multi-Hop RAG Workflow

number of retrieval steps. Although these methods advance multi-hop reasoning, they suffer from high resource consumption, as shown in Pattern (a) of Figure 1. In contrast, we treat each sub-query reasoning step as an independent task and dynamically select models based on sub-task complexity and retrieval confidence. This balances computational efficiency and accuracy, as illustrated in Pattern (b).

3.2 Design and Implementation of RAG-on-a-Diet

Our agent is loosely coupled with the multi-hop RAG pipeline. It builds a reasoning state from sub-queries and retrieved information, makes policy decisions through an RL module optimized by reward feedback, and outputs actions to guide the RAG system in selecting a suitable model from the LLM pool, as illustrated in Figure 2.

This process resembles an "intelligent nutritionist" providing RAG with dynamic, personalized "meal preparation" based on real-time needs.

3.3 Reward Function Design

We design a multi-component reward function to train RAG-on-a-Diet effectively for multi-hop RAG:

$$R_t = \begin{cases} R_{\text{final}} + \alpha R_{\text{step}} + \beta C_t + \delta R_{\text{balance\_reward}} & t = T \\ R_{\text{cumulative}} + \alpha R_{\text{step}} + \beta C_t + \delta R_{\text{balance\_reward}} & t \neq T \end{cases} \quad (1)$$

where T is the total number of hops, and α, β, δ are validation-tuned hyperparameters balancing quality and cost. The components are:

- C_t: Cost overhead of hop t, converted to a negative reward (lower cost $\rightarrow$ higher reward);
- R_{step}: Step-wise accuracy reward for the current hop;
- $R_{\text{cumulative}}$: Cumulative accuracy from hop 1 to t (for $t < T$), encouraging consistent intermediate quality;
- R_{final}: Final-answer accuracy reward, applied only at $t = T$;
- $R_{\text{balance\_reward}}$: Quality-per-unit-cost reward, promoting efficiency.

3.3.1 Step Cost

There are two cost estimation schemes, which we analyze and compare in detail in Appendix A. Given that most related work adopts token cost simplification, e.g., (Lewis et al., 2020; Chen et al., 2024; Bai et al., 2023; Albatarni et al., 2024), this paper uses the token-and-cloud-service-invocation pricing model for cost calculation.

3.3.2 Step Reward

Provides immediate feedback at each hop to encourage effective local reasoning:

$$R_{\text{step}}(t) = w_1 \cdot M(\text{Sub}Q_t, \text{Retrieval}_t) + w_2 \cdot \Delta C(t) \quad (2)$$

$M(\text{Sub}Q_t, \text{Retrieval}_t)$ is the sub-query–retrieval matching score, $\Delta C(t) = \text{Conf}(\text{relation}_t) - \text{Conf}(\text{relation}_{t-1})$ tracks confidence gain in reasoning relations. We set $w_1 = 0.6$, $w_2 = 0.4$ by default, adjustable for task-specific tuning.

3.3.3 Cumulative Reward

To balance step-wise dense and terminal sparse rewards, we introduce a cumulative reward computed at each step t based on the cumulative quality of reasoning steps up to t:

$$R_{\text{cumulative}}(t) = \begin{cases} A\left(Q, \{\text{Ans}_k \cdot M_k\}_{k=1}^{t}\right) & t \neq T \\ 0 & t = T \end{cases} \quad (3)$$

where Q is the original problem, Ans_k is the answer at step k, $M_k = M(\text{SubQ}_k, \text{Retrieval}_k)$ is the Sub-query-retrieval matching score, and $A(\cdot)$ evaluates global accuracy.

3.3.4 Final Reward

To align training with task performance, the final reward is assigned only at step $t = T$ and combines answer quality with reasoning consistency:

$$R_{\text{final}} = \text{F1}(\text{FinalAns}) + \lambda \cdot \frac{1}{T} \sum_{t=1}^{T-1} R_{\text{cumulative}}(t) \quad (4)$$

$\text{F1}(\text{FinalAns}) \in [-2, 1]$ reflects the final answer quality, and λ balances final accuracy against cumulative reasoning quality, tuned empirically.

3.3.5 Balance Reward

A balanced reward is introduced, using quality gain per unit compute cost to jointly incentivize answer quality and cost efficiency. The full reward combines R_{final} for final correctness, $R_{\text{cumulative}}$ for multi-step consistency, and R_{step} for immediate decision quality, thus achieving a three-layer balance of **instantaneity**, **cumulativeness**, and **decisiveness**. This mitigates issues from sparse terminal rewards or overemphasis on process, while explicitly optimizing the quality–cost trade-off.

3.4 Credit Allocation Optimization and Generalization Reasoning Guarantee

To ensure precise credit assignment and robust generalization in multi-hop reasoning, we propose two complementary strategies in our reward design.

3.4.1 Refined Credit Allocation

We enforce precise credit assignment via a three-layer mechanism:

- **Local**: R_{step} ties each action's reward to its immediate retrieval quality and relation confidence, preventing final-outcome monopolization of credit.
- **Cumulative**: $R_{\text{cumulative}}$ propagates credit across reasoning steps, ensuring early correct actions in long chains receive feedback.
- **Global**: R_{final} calibrates the entire trajectory toward the correct final answer.

Credit traceability is further enhanced by logging action–reward mappings during training.

3.4.2 Generalization Guarantees Against Spurious Priors

To discourage fitting superficial cues (e.g., entity frequency, fixed reasoning paths), we:

- Use only reasoning-relevant features (e.g., relation confidence, matching score), excluding surface statistics;
- Embed a quality–cost trade-off via $R_{\text{balance\_reward}}$, penalizing strategies that exploit dataset biases (e.g., blindly invoking max models);
- Pre-train on de-biased expert data, filtering out rule-based decisions driven by priors (e.g., max-model calls triggered solely by entity count).

Together, these ensure the controller learns robust, generalizable reasoning rather than dataset-specific artifacts.

3.5 Explicit Termination Mechanism for Multi-Hop Reasoning

Multi-hop reasoning suffers from error accumulation, where accuracy drops to 60% of the random baseline beyond 5 hops and causes wasted computation. Unlike methods like AT-RAG (Rezaei et al., 2024), which lack explicit stopping criteria, we implement an empirically derived termination policy grounded in observed error accumulation patterns based on empirical error patterns.

1. **Hop Limit**: Enforce a hard cap of 5 hops to avoid accuracy collapse from over-iteration. This constraint aligns with prior work showing diminishing returns and error propagation in long-chain reasoning.
2. **Confidence Check**: Before each new hop, compute the current answer confidence $\text{conf}(a_t)$. If $\text{conf}(a_t) < \tau$ (with $\tau = 0.3$ tuned on the validation set), terminate early to avoid propagating errors.

This mechanism is computationally efficient (no extra training) and interpretable—its parameters stem directly from observed performance trends. Unlike AT-RAG's fixed-step stopping, it adaptively halts unproductive reasoning, preserving accuracy while saving resources.

3.6 Agent Training

Following (Macuglia et al., 2025; Luo et al., 2025b), we adopt a two-stage training paradigm:

Behavior Cloning (BC) pre-training establishes a stable policy, followed by Proximal Policy Optimization (PPO) fine-tuning to boost reinforcement learning efficiency and avoid local optima. The full process consists of three phases (see Figure 3):

- **Phase 1**: A rule-based expert system, which is enhanced by a hierarchical narrowing prediction algorithm and human expertise, generates high-quality demonstrations on HotpotQA. For further details, please refer to appendix B.2.
- **Phase 2**: The agent learns from expert trajectories via BC, establishing a strong initialization; negative samples are collected to quantify behavioral deviation.
- **Phase 3**: PPO fine-tuning uses our multi-dimensional reward to iteratively refine the policy, enabling the agent to surpass the expert rules through environment interaction.

This "imitate-then-optimize" strategy ensures robust and efficient training, analogous to learning from data and experience first, then refining skills through practice.

4 Experimental Setting

4.1 Training Data

We use the multi-hop QA dataset HotpotQA (Yang et al., 2018), selecting samples with diverse reasoning difficulty and input lengths:

- **Difficulty**: Easy, medium, and hard, based on decomposition complexity and required reasoning steps;
- **Input Length**: Short, medium-long, and long texts;

4.2 Experimental Setup

HotpotQA (Yang et al., 2018),MuSiQue (Trivedi et al., 2022) and 2wikimultihop (Ho et al., 2020) serves as the test set. We implement RAG using RAGFlow 0.19 and use Qwen3-30B for question decomposition. Three model scales are deployed for query processing: DS-R1-671B (max), Qwen3-30B (middle), and Qwen3-4B (min).

4.3 Ablation Test

Following the feature ablation framework of (Leontjeva and Kuzovkin, 2016), we evaluate the contribution of static and temporal features to routing decisions via three ablation groups: (1) with key static feature subsets removed, (2) with key temporal feature subsets removed, and (3) with entire feature types ablated. This reveals whether static and temporal features jointly enhance performance.

4.4 Comparison Methods

We compare RAG-on-a-Diet against four baselines on reasoning quality (F1) and computational cost, under identical setups:

- **Standard RAG** (Lewis et al., 2020): Single-step retrieval with DS-R1 as generator.
- **IRCoT** (Trivedi et al., 2023): Iterative retrieval with a maximum of 5 steps; tested with large, medium, and small generators.
- **Adaptive-RAG** (Jeong et al., 2024): Uses a classifier to choose single- or multi-step retrieval; employs DS-R1.
- **SOTA Baselines**: Router-R1, FrugalRAG, GlobalRAG (Luo et al., 2025a), and DualRAG,covering advanced routing, cost-efficient, and global optimization paradigms.
- **Expert Rule-Based**: uses the same static/temporal features as the RL agent, ensuring fair comparison of policy sophistication (not feature advantage).
- **RAG-on-a-Diet (Ours)**: BC+PPO-trained agent with multi-dimensional reward (final, cumulative, stepwise, cost, balance); dynamic query routing to Qwen3-4B/30B/DS-R1-671B; heuristic explicit stopping (5-hop upper bound + $\tau = 0.3$).

5 Results and Discussion

5.1 Main Results

To comprehensively evaluate our hop-level dynamic LLM allocation framework, we compare RAG-on-a-Diet with state-of-the-art baselines on four multi-hop QA benchmarks, including Router-R1, FrugalRAG, GlobalRAG, and DualRAG. As shown in Table 1, our method achieves a Pareto-optimal frontier, retaining near-SOTA accuracy while reducing costs by up to 60.07% relative to full max-model baselines. Detailed test results are presented in Table 1.

Comparison on HotpotQA:

- *vs. IRCoT (DS-R1-671B):* RAG-on-a-Diet only incurs a 3.27% F1 drop on train/val sets and 3.70% on test sets, while reducing costs by 59.73% and 60.07%, respectively.

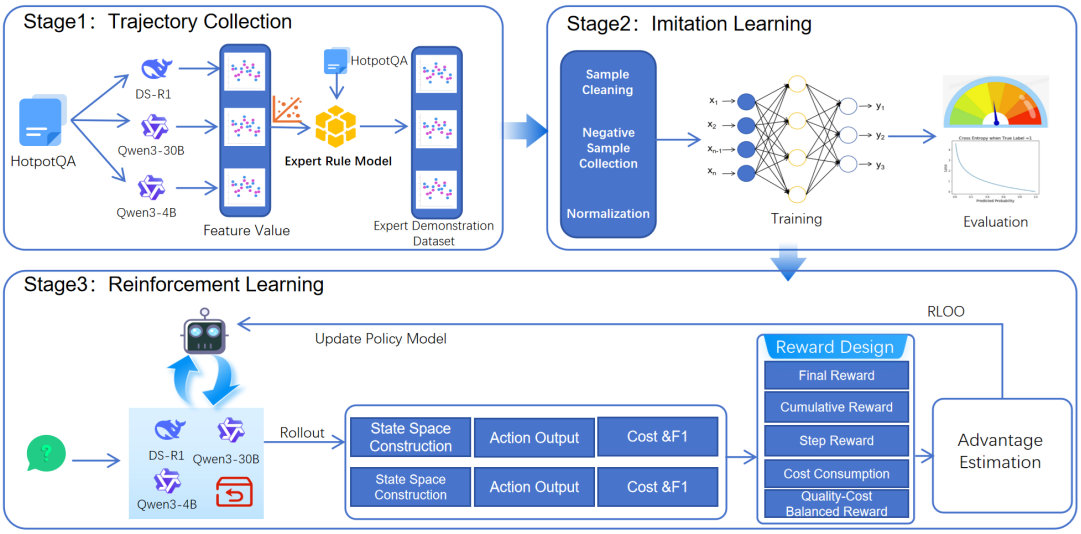

Figure 3: Three Phases of RAG-on-a-Diet Training. Phase 1: Rule-based expert with hierarchical narrowing generates demonstrations. Phase 2: Behavior cloning from expert data with deviation analysis. Phase 3: PPO fine-tuning using multi-dimensional rewards to exceed expert performance.

- *vs. Adaptive-RAG (DS-R1-671B):* It achieves nearly identical F1 on train/val (33.59% cost reduction) and a slight F1 gain on test, with a 37.30% cost cut.

- *vs. SOTA Baselines:* GlobalRAG attains the highest raw F1 (0.7612) via global planning but at near-IRCoT-level cost. RAG-on-a-Diet achieves 97% of GlobalRAG's F1 at only 45% of its cost, validating that dynamic hop-level routing eliminates redundant max-model inference without sacrificing performance.

- *vs. Expert Rule Model:* RAG-on-a-Diet outperforms it on test F1 (0.7387 vs. 0.7332) with lower computational costs, demonstrating a superior quality-efficiency trade-off.

These results validate our framework's core objective: the reinforcement learning agent effectively trades small F1 loss (up to 3.70%) for large cost savings (up to 60.07%), confirming the feasibility of RAG-on-a-Diet's design concept.

Generalization on MuSiQue and 2Wiki: Consistent trends are observed on MuSiQue and 2Wiki datasets. RAG-on-a-Diet not only maintains competitive F1 scores comparable to all baselines but also surpasses the carefully human-engineered Expert Rule Model in F1, while consistently keeping costs below those of IRCoT, Adaptive-RAG, and all SOTA baselines. This further verifies the strong generalization capability of our agent across diverse multi-hop reasoning tasks and demonstrates the powerful creativity of reinforcement learning.

5.2 Sensitivity Analysis of Reward Weights

The reward function in Equation 1 contains three key hyperparameters: α (step-wise accuracy), β (cost penalty), and δ (quality-cost balance). We conduct a sensitivity analysis by varying one parameter while fixing the other two to assess their impact on convergence and the F1–cost Pareto frontier. As shown in Figure 4:

- **Step-wise Weight (α):** Increasing α from 0.4 to 1.6 improves early-hop retrieval precision and reduces error propagation. However, $\alpha > 1.4$ causes over-cautious behavior, triggering max models even for simple queries.

- **Cost Penalty (β):** The agent is highly sensitive to β. Low β (≤ 0.3) leads to exclusive use of DS-R1, while high β (≥ 0.8) forces overuse of Qwen3-4B, causing a 14% F1 drop on hard samples.

- **Balance Factor (δ):** δ stabilizes training. Setting $\delta \in [0.5, 0.8]$ encourages "high-efficiency" hops and yields the best accuracy–cost trade-off.

This confirms our framework offers tunable control over the accuracy–cost balance under varying hardware constraints. We further conduct an incremental reward ablation on HotpotQA, with results in Table 2. Starting from a terminal-only baseline, we sequentially add each reward component, observing consistent improvements in both accuracy

Methods	Gen. Model	HpQA(Train/Val)		HpQA (Test)		MuSiQue		2Wiki		Avg. F1/Cost
		F1	Cost	F1	Cost	F1	Cost	F1	Cost	
Standard RAG	DS-R1-671B	0.6576	14.45	0.6703	15.34	0.6045	14.88	0.5944	12.35	0.04449
IRCoT	DS-R1-671B	0.7514	42.12	0.7671	43.46	0.7171	46.96	0.7266	37.54	0.01753
	Qwen3-30B	0.6685	17.12	0.7133	17.77	0.6783	20.49	0.6710	15.56	0.03885
Adaptive-RAG	DS-R1-671B	0.7291	25.54	0.7315	27.67	0.6789	28.66	0.6918	24.89	0.02662
Router-R1	Qwen3-4B/Q3-30B/DS-R1	0.7510	34.76	0.7538	35.22	0.7089	37.84	0.7157	31.46	0.02112
FrugalRAG	DS-R1-671B	0.7371	26.66	0.7412	28.93	0.6953	31.27	0.7023	25.88	0.02566
GlobalRAG	DS-R1-671B	0.7487	36.94	0.7612	38.71	0.7198	42.15	0.7234	34.29	0.01953
DualRAG	Qwen3-4B/DS-R1	0.6989	25.17	0.7156	25.87	0.6689	27.82	0.6795	22.53	0.02741
Expert Rule Model	Qwen3-4B/Q3-30B/DS-R1	0.7435	28.71	0.7332	30.11	0.7070	29.51	0.7009	23.88	0.02589
RAG-on-a-Diet	Qwen3-4B/Q3-30B/DS-R1	0.7268	16.96	0.7387	17.35	0.7042	20.88	0.7106	16.18	0.04077

Table 1: Performance Comparison on Core Benchmarks. RAG-on-a-Diet achieves a *Pareto-optimal* frontier across three multi-hop QA datasets. Compared to state-of-the-art baselines, our method maintains near-SOTA F1 scores (e.g., 0.7387 on HotpotQA Test) while delivering up to 60.07% inference cost reduction.

Reward Configuration	Test F1	Test Cost	Avg. Hops	QPC (F1/Cost)
Terminal-Only ($R_{\text{final}} + \beta C_t$)	0.6985	23.72	4.1	0.0294
+ R_{step}	0.7194	19.86	3.6	0.0362
+ $R_{\text{cumulative}}$	0.7301	18.53	3.4	0.0394
Full (all components)	0.7387	17.35	3.2	0.0426

Table 2: Incremental reward component ablation on HotpotQA.

and efficiency. The full reward function achieves +4.0 F1 and -26.9% cost vs. the terminal-only baseline, confirming effective credit allocation.

5.3 Quantifying Agent Overhead and Latency

To assess the overhead of dynamic routing, we measure the end-to-end latency and computational cost of the RAG-on-a-Diet agent against base LLMs. Our lightweight RL agent has only 150K parameters and runs efficiently on CPU, with a per-hop decision latency of 9.05 ms—three orders of magnitude lower than 3B-parameter routers and just 0.11% of DS-R1's latency. Our system also eliminates model switching overhead via vLLM preloading. Detailed latency, scale, and speedup results are provided in Appendix D.

Feature extraction is embedded in retrieval metadata parsing without extra API calls. The agent thus introduces negligible overhead while achieving substantial time and cost savings.

5.4 Ablation Test Results

Before presenting detailed single-feature ablation results, we first verify the noise robustness of our multi-feature design via feature dropout experiments with probabilities $p = 0.1, 0.2, 0.3$. Our model maintains stable performance even under moderate feature perturbation, with full results and analysis in Appendix E.

To systematically analyze input feature roles, we conduct ablation studies across static, temporal, and combined feature groups (see Appendix F). Results show that: (1) **Static features** (e.g., entity count, question length, entity importance, relation salience) are all effective: entity-related features govern cost control, while question length and relation salience directly improve reasoning accuracy; (2) **Temporal features** (e.g., global quality, global confidence) are critical for dynamic cost regulation—their removal leads to significant cost inflation, with global confidence especially vital for F1 retention; (3) **Feature combinations** show clear synergy: ablating static pairs (e.g., entity count + importance) severely degrades quality, while ablating temporal pairs (e.g., global quality + confidence) mainly destabilizes cost.

This confirms that static features provide prior-based judgment, while temporal features supply in-process feedback, making them complementary for adaptive efficiency.

5.5 Effectiveness of Explicit Heuristic Termination Mechanism

To verify our explicit heuristic termination mechanism, we compare it with two baselines:

1. No-Term: No active termination (stops only at 10 hops, simulating AT-RAG).

2. Hop-Only: Only 5-hop hard bound (no confidence pre-check).

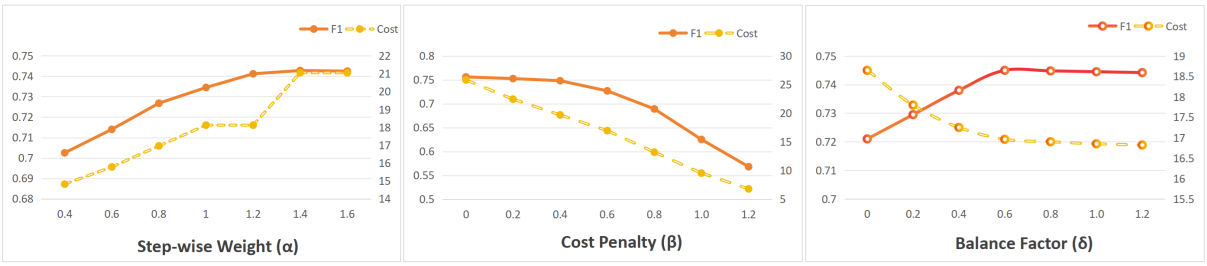

Figure 4: Sensitivity Analysis of Reward Weights:
$\alpha \in [0.8, 1.2]$ achieves optimal trade-off; gains saturate beyond 1.4. β is negatively correlated with both F1 and cost; an inflection point occurs at $\beta = 0.6$, beyond which F1 drops sharply. δ rapidly improves F1 and reduces cost before plateauing, confirming its stabilizing role.

5.5.1 Setup and Metrics

Experiments are on multi-hop benchmarks and RAG-on-a-Diet test set, with 4 core metrics: Reasoning Accuracy (F1), Avg. Iteration Hops, Avg. Call Cost, and Error Accumulation Rate (lower = severer error accumulation).

5.5.2 Results and Analysis

Table 3 summarizes the performance of the three methods. Key conclusions (proving our heuristic termination is superior to baselines):

- Our method suppresses error accumulation effectively (72.5% rate vs. 38.1% for No-Term, 60.2% for Hop-Only) and achieves the highest F1 (0.7387, 13.2% higher than No-Term, 7.2% higher than Hop-Only).
- It reduces invalid iterations (3.2 avg. hops vs. 8.7 for No-Term, 4.9 for Hop-Only) and cuts cost significantly (17.35 vs. 38.46 for No-Term, 22.89 for Hop-Only), avoiding resource waste.
- Dual-rule synergy (hop constraint + confidence check) outperforms single constraint, verifying the superiority of our heuristic termination mechanism.

5.5.3 Credit Allocation and Reasoning Generalization

Building on the ablation results in Section 5.4, we draw three high-level conclusions on our framework's design and generalization.

First, sensitivity to temporal features (e.g., *global quality*) confirms our cumulative credit allocation mechanism effectively translates dynamic signals into resource-aware decisions, mitigating redundant computation.

Second, dependence on static features such as *entity importance* shows that grounded prior cues resolve reward ambiguity in multi-hop RL, enabling stable cost-accuracy trade-offs.

Third, consistent gains over expert rule baselines across diverse datasets verify the agent learns transferable reasoning strategies rather than dataset-specific biases, validating its generalizable control.

5.6 Discussion

The results reveal three core advantages of RAG-on-a-Diet: (1) It breaks the "cost reduction harms performance" trade-off with only 3.7% F1 drop vs. IRCoT and higher F1 than the Expert Rule Model (0.7387 vs. 0.7332), verifying RL agents outperform handcrafted heuristics; (2) It achieves universal cost savings (60.07% over IRCoT, 37.30% over Adaptive-RAG, 42.4% over the expert model), showing robust dynamic allocation; (3) It lifts cost-effectiveness by 2.41× (vs. IRCoT) and 1.61× (vs. Adaptive-RAG), lowering deployment barriers.

Ablation studies further validate these gains: ablating *entity count* raises cost by 27.4%(form 17.35 to 22.10) due to excessive max-model use (confirming its role in complexity-aware routing); Removing *global quality* or *global confidence* degrades cost by 29.8% (from 17.35 to 22.51) or F1 by 2.4% (from 0.7387 to 0.7207), respectively, which highlights the necessity of our three-layer reward design. These studies also clarify feature correlations with quality, cost, and their balance, providing key references for subsequent task optimization.

Though evaluated on multi-hop QA, RAG-on-a-Diet's core principle (hop-wise RL control with multi-dimensional reward) is task-agnostic, as long as reasoning can be decomposed into sequential sub-steps with measurable confidence and cost.

5.7 Challenge Set Experiments

To evaluate robustness beyond in-distribution benchmarks, we test on three challenging out-of-

Method	Reasoning Accuracy (F1)	Avg. Iteration Hops	Avg. Cost	Error Accumulation Rate (5 hops)
No-Term	0.6523	8.7	38.46	38.1%
Hop-Only	0.7017	4.9	22.89	60.2%
Ours	**0.7387**	**3.2**	**17.35**	**72.5%**

Table 3: Performance of Explicit Heuristic Termination Mechanism. Lower Error Accumulation Rate means faster error accumulation with more hops; Avg. Call Cost = quantified API/computational overhead.

Type	Quality	Cost	Balance
Static	QLen	ECnt	ECnt, Imp, RS
Temporal	GConf	GQual	CQ, GSal
Combined	ECnt+Imp	GQ+Cnf	PHQ+Cnf

Table 4: Feature impact (QLen=Question Length, ECnt=Entity Count, Imp=Importance, RS=Relation Salience, GConf=Global Confidence, GQual=Global Quality, CQ=Current Quality, GSal=Global Salience, PHQ=Previous-Hop Quality, Cnf=Confidence).

Method	Simple F1↑	Simple MIC↓	Simple QPC↑	FalsePrem F1↑	FalsePrem MIC↓	FalsePrem QPC↑
IRCoT + DS-R1	0.8234	38.12	0.0216	0.3156	41.78	0.0076
Router-R1	0.8156	28.45	0.0287	0.3289	33.56	0.0098
FrugalRAG	0.7989	22.67	0.0352	0.3078	26.34	0.0117
GlobalRAG	0.8198	35.89	0.0228	0.3312	39.45	0.0084
DualRAG	0.7834	20.56	0.0381	0.2867	24.34	0.0118
Expert Rule	0.8023	16.78	0.0478	0.3023	22.56	0.0134
RAG-on-a-Diet	0.8067	9.12	0.0885	0.3189	11.23	0.0284

Table 5: CRAG Subset Results

Method	Simple Avg Hops	Simple Action % (S/M/L)	FalsePrem Avg Hops	FalsePrem Action % (S/M/L)
IRCoT + DS-R1	3.0	0/0/100	3.0	0/0/100
FrugalRAG	2.2	0/0/100	2.4	0/0/100
RAG-on-a-Diet	1.9	68/22/10	1.5	52/29/19

Table 6: CRAG Action Proportion & Termination Behavior

Method	F1↑	MIC↓	QPC↑
IRCoT + DS-R1	0.6845	41.23	0.0166
Router-R1	0.6723	33.89	0.0198
FrugalRAG	0.6612	27.56	0.0240
GlobalRAG	0.6934	37.45	0.0185
DualRAG	0.6412	24.67	0.0260
Expert Rule	0.6578	27.89	0.0236
RAG-on-a-Diet	0.6589	17.34	0.0380

Table 7: Bamboogle Results (125 hard compositional 2-hop questions, zero-shot)

distribution datasets: CRAG-Simple, CRAG-False Premise, and Bamboogle, covering factoid accuracy, false premise detection, and adversarial multi-hop reasoning.

5.7.1 CRAG Benchmark Evaluation

We evaluate on the Comprehensive RAG Benchmark (CRAG), which includes two subsets:

- **CRAG-Simple**: Factoid questions with distractor documents.
- **CRAG-False Premise**: Questions with invalid presuppositions requiring rejection.

Table 5 reports F1, MIC, and QPC for 7 methods. Table 6 analyzes hop count and routing behavior.

On CRAG-Simple, our method terminates at 1.9 hops (68% min-model routing), reducing MIC by 76.1% vs. IRCoT. On CRAG-False Premise, all methods achieve 0.30 F1, but our termination cuts IRCoT's cost by 73.1%, outperforming FrugalRAG's 37.0% savings.

5.7.2 Bamboogle Benchmark Evaluation

We further test on Bamboogle, an adversarial dataset of 125 hard 2-hop questions for zero-shot generalization,with results shown in Table 7.

While GlobalRAG attains the highest F1 (0.6934) on hard queries, our method maintains competitive accuracy with the maximum QPC (0.0380), 57.9% MIC savings over IRCoT, and robust cost-efficiency on uniformly hard tasks.

6 Conclusion

We propose RAG-on-a-Diet, an RL-based dynamic routing framework for compute-efficient multi-hop RAG. Modeling reasoning as an MDP with a three-tier reward (step-wise, cumulative, final), our approach selects the optimal model per hop and terminates low-confidence paths early. Experiments on HotpotQA, MuSiQue, and 2WikiMultiHopQA demonstrate consistent advantages: on HotpotQA, it cuts cost by 60.1% with only a 3.7% F1 drop against IRCoT, while surpassing handcrafted expert systems; comparable gains appear on MuSiQue and 2Wiki. Ablations verify the necessity of feature-aware routing and multi-dimensional reward design. Future work will extend to conversational and multimodal RAG, strengthen long-chain reasoning, and optimize real-world deployment.

Limitations

This work has three main limitations. First, RAG-on-a-Diet is currently validated only on static multi-hop QA tasks, lacking optimization for dynamic conversational or multimodal RAG scenarios. Second, like other LLM-based systems, it may produce hallucinations, requiring external validation for safety-critical use. Finally, while our cost analysis assumes ideal single-user conditions to verify trend consistency, it does not account for industrial factors such as high-concurrency scheduling, batch inference optimization, or hardware power consumption; future work will refine the cost model with real-world parameters and deployment data.

Acknowledgments

We would like to thank all our reviewers for their insightful comments, substantial discussions and constructive suggestions. This work was partially supported by the National Natural Science Foundation of China (No. 62476125), the Noncommunicable Chronic Diseases–National Science and Technology Major Project (No. 2025ZD0550704), the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (No. JYB2025XDXM118), Nanjing University International Collaboration Initiative, and the Rhino-Bird Middle School Science Research Training Program (NO.16851).

References

Salam Albatarni, Sohaila Eltanbouly, and Tamer Elsayed. 2024. Graded Relevance Scoring of Written Essays with Dense Retrieval. In *Proc. SIGIR'24*, pages 1329–1338. ACM.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *Proc. ICLR'24*. OpenReview.net.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, and 29 others. 2023. Qwen Technical Report. *CoRR*, abs/2309.16609.

Jihwan Bang, Juntae Lee, Seunghan Yang, and Sungha Choi. 2025. Think Straight, Stop Smart: Structured Reasoning for Efficient Multi-Hop RAG. *CoRR*, abs/2510.19171.

Ziyi Cao, Yunhe Xie, Bingquan Liu, and Kun Bu. 2025. Re3mhqa: Retrieve, remove, and return facts in multi-hop QA. *Expert Syst. Appl.*, 281:127566.

Lingjiao Chen, Matei Zaharia, and James Zou. 2024. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *Trans. Mach. Learn. Res.*, 2024.

Rong Cheng, Jinyi Liu, Yan Zheng, Fei Ni, Jiazhen Du, Hangyu Mao, Fuzheng Zhang, Bo Wang, and Jianye Hao. 2025. DualRAG: A Dual-Process Approach to Integrate Reasoning and Retrieval for Multi-Hop Question Answering. In *Proc. ACL'25*, pages 31877–31899. Association for Computational Linguistics.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. In *Proc. ICLR'24*. OpenReview.net.

Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian del Carmen Hipolito Garcia, Menglin Xia, Laks V. S. Lakshmanan, Qingyun Wu, and Victor Rühle. 2025. BEST-Route: Adaptive LLM Routing with Test-Time Optimal Compute. In *Proc. ICML'25*, Proceedings of Machine Learning Research. PMLR / OpenReview.net.

Xinyan Guan, Jiali Zeng, Fandong Meng, Chunlei Xin, Yaojie Lu, Hongyu Lin, Xianpei Han, Le Sun, and Jie Zhou. 2025. DeepRAG: Thinking to Retrieval Step by Step for Large Language Models. *CoRR*, abs/2502.01142.

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. *Nat.*, 645(8081):633–638.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval Augmented Language Model Pre-Training. In *Proc. ICML'20*, Proceedings of Machine Learning Research, pages 3929–3938. PMLR.

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 *Proc. COLING'20*, pages 6609–6625. International Committee on Computational Linguistics.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *ACM Trans. Inf. Syst.*, 43(2):42:1–42:55.

Abhinav Java, Srivathsan Koundinyan, Nagarajan Natarajan, and Amit Sharma. 2025. FrugalRAG: Learning to retrieve and reason for multi-hop QA. *CoRR*, abs/2507.07634.

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 *Proc. NAACL'24*, pages 7036–7050. 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 *Proc. EMNLP'23*, pages 7969–7992. Association for Computational Linguistics.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed Prompting: A Modular Approach for Solving Complex Tasks. In *Proc. ICLR'23*. OpenReview.net.

Moshe Lahmy and Roi Yozevitch. 2025. Replace, Don't Expand: Mitigating Context Dilution in Multi-Hop RAG via Fixed-Budget Evidence Assembly. *CoRR*, abs/2512.10787.

Anna Leontjeva and Ilya Kuzovkin. 2016. Combining Static and Dynamic Features for Multivariate Sequence Classification. In *Proc. DSAA'16*, pages 21–30. IEEE.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33 (NeurIPS'20)*.

Xin-Yi Li, Wei-Jun Lei, and Yu-Bin Yang. 2023. From Easy to Hard: Two-Stage Selector and Reader for Multi-Hop Question Answering. In *Proc. ICASSP'23*, pages 1–5. IEEE.

Zhonghao Li, Kunpeng Zhang, Jinghuai Ou, Shuliang Liu, and Xuming Hu. 2025. TreeHop: Generate and Filter Next Query Embeddings Efficiently for Multi-hop Question Answering. *CoRR*, abs/2504.20114.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's Verify Step by Step. In *Proc. ICLR'24*. OpenReview.net.

Jinchang Luo, Mingquan Cheng, Fan Wan, Ni Li, Xiaoling Xia, Shuangshuang Tian, Tingcheng Bian, Haiwei Wang, Haohuan Fu, and Yan Tao. 2025a. GlobalRAG: Enhancing Global Reasoning in Multi-hop Question Answering via Reinforcement Learning. *CoRR*, abs/2510.20548.

Pengcheng Luo, Yunyang Zhao, Bowen Zhang, Genke Yang, Boon-Hee Soong, and Chau Yuen. 2025b. SABR: A Stable Adaptive Bitrate Framework Using Behavior Cloning Pretraining and Reinforcement Learning Fine-Tuning. *CoRR*, abs/2509.10486.

Maël Macuglia, Paul Friedrich, and Giorgia Ramponi. 2025. Fine-tuning Behavioral Cloning Policies with Preference-Based Reinforcement Learning. *CoRR*, abs/2509.26605.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and Narrowing the Compositionality Gap in Language Models. In *Findings of EMNLP'23*, Findings of ACL, pages 5687–5711. Association for Computational Linguistics.

Mohammad Reza Rezaei, Maziar Hafezi, Amit Satpathy, Lovell Hodge, and Ebrahim Pourjafari. 2024. ATRAG: An Adaptive RAG Model Enhancing Query Efficiency with Topic Filtering and Iterative Reasoning. *CoRR*, abs/2410.12886.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025a. R1-Searcher: Incentivizing the Search Capability in LLMs via Reinforcement Learning. *CoRR*, abs/2503.05592.

Huatong Song, Jinhao Jiang, Wenqing Tian, Zhipeng Chen, Yuhuan Wu, Jiahao Zhao, Yingqian Min, Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025b. SmartSearcher: Incentivizing the Dynamic Knowledge Acquisition of LLMs via Reinforcement Learning. In *Findings of EMNLP'25*, pages 13572–13586. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop Questions via Single-hop Question Composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. In *Proc. ACL'23*, pages 10014–10037. Association for Computational Linguistics.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. In *Proc. ACL'24*, pages 9426–9439. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35 (NeurIPS'22)*.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective Retrieval Augmented Generation. *CoRR*, abs/2401.15884.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proc. EMNLP'18*, pages 2369–2380. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *Proc. ICLR'23*. OpenReview.net.

Haozhen Zhang, Tao Feng, and Jiaxuan You. 2025. Router-R1: Teaching LLMs Multi-Round Routing and Aggregation via Reinforcement Learning. *CoRR*, abs/2506.09033.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. DeepResearcher: Scaling Deep Research via Reinforcement Learning in Real-world Environments. In *Proc. EMNLP'25*, pages 414–431. Association for Computational Linguistics.

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 *Proc. ICLR'23*. OpenReview.net.

A Cost Quantification Scheme

A.1 Cost Quantification Objective

To address the "quality-cost balance" requirement of RAG, this study considers two cost quantification schemes:

1. **Market Token-based Cost**: A normalized cost model is constructed based on the public LLM token pricing standards of mainstream cloud service providers, which serves as the core cost input for the agent's reward function.

2. **Real Deployment Cost**: The actual cost of single-machine private deployment is derived from the inference latency measured on the laboratory's in-house H200 hardware, which is used to compare and analyze the engineering practicality of the token-based cost model.

This appendix elaborates on the calculation logic and comparative results of the two cost metrics, clarifying the design basis and application value of the cost quantification schemes.

A.2 Core Definitions and Unified Experimental Premises

Model Selection: Three types of open-source LLMs with different parameter scales, namely Qwen3-4B (min model), Qwen3-30B (middle model) and DS-R1-671B (max model), are selected to cover the cost difference range from lightweight to max-scale models.

Definition of Inference Task: For a single-round RAG inference task, the input token count is fixed as $T_{\text{in}} = 2000$ (including user query + retrieved knowledge snippets), and the output token count is fixed as $T_{\text{out}} = 512$ (standard response length), to ensure consistency in cost calculation.

Hardware and Optimization Configuration: The H200 is deployed with exclusive single-card access, using FP8 quantization precision + vLLM inference framework optimization. Multi-user concurrent scheduling is disabled to ensure a linear correlation between inference latency and hardware resource consumption.

Hardware Pricing Benchmark: Referring to enterprise-level hardware depreciation standards, the hourly depreciation cost of a single H200 card is 1.2 CNY, which is used for the monetization conversion of real deployment costs.

A.3 Cloud Service-Based Token Invocation Cost Quantification Scheme

The market token unit prices are obtained from the public API quotations of mainstream cloud service providers (Table 8). Taking the smallest-parameter model Qwen3-4B as the baseline, the relative cost coefficients of other models are calculated to enable normalized comparison of costs across models. A single-round Token cost consists of input cost and output cost, with the calculation formulas as follows:

$$\text{Input Cost} = m_{\text{in}} \cdot \frac{T_{\text{in}}}{10^6} \cdot P_{\text{in-base}}$$

$$\text{Output Cost} = m_{\text{out}} \cdot \frac{T_{\text{out}}}{10^6} \cdot P_{\text{out-base}}$$

$$C_{\text{norm}} = \text{Input Cost} + \text{Output Cost}$$

where:

- $P_{\text{in-base}} = 0.15$ CNY/mtokens, $P_{\text{out-base}} = 0.6$ CNY/mtokens, which are the base Token unit prices of Qwen3-4B;
- m_{in} and m_{out} are the relative cost coefficients in Table 8, used to quantify the cost differences between different models.

A.3.1 Model-Specific Calculation Results

- **Qwen3-4B**:

$$\begin{aligned} C_{\text{norm}} &= 1 \times \frac{2000}{10^6} \times 0.15 + 1 \times \frac{512}{10^6} \times 0.6 \\ &= 0.0006072 \approx 0.00061 \text{ CNY} \end{aligned}$$

- **Qwen3-30B**:

$$C_{\text{norm}} = 13 \times 0.0006072 \approx 0.0079 \text{ CNY}$$

- **DS-R1-671B**:

$$C_{\text{norm}} = 26 \times 0.0006072 \approx 0.0158 \text{ CNY}$$

A.3.2 Normalized Cost Ratio

The Token cost ratio of the three models is: Qwen3-4B : Qwen3-30B : DS-R1-671B = 1 : 13 : 26

A.4 Private Deployment-Based Hardware Amortization Cost Quantification Scheme

A.4.1 Hardware Cost Conversion

The hourly depreciation cost of a single H200 card is converted to the unit cost per second:

$$\begin{aligned} \text{Unit Cost per Second} &= \frac{1.2 \text{ CNY}}{3600 \text{ s}} \\ &\approx 0.000333 \text{ CNY/s} \end{aligned}$$

A.4.2 H200 Measured Single-Round Inference Latency

Based on the measured data optimized with FP8 precision and vLLM, the single-round inference latencies of the three models are:

- Qwen3-4B: 0.1 s (100 ms)
- Qwen3-30B: 1.5 s (1500 ms)
- DS-R1 (671B): 8.3 s (8300 ms)

A.4.3 Single-Round Real Cost Formula

$C_{\text{real}} = \text{Unit Cost per Second} \times \text{Single-Round Latency}$

A.4.4 Model-Specific Calculation Results

- **Qwen3-4B**: $C_{\text{real}} = 0.000333 \times 0.1 = 0.0000333 \approx 0.000033$ CNY
- **Qwen3-30B**: $C_{\text{real}} = 0.000333 \times 1.5 = 0.0004995 \approx 0.0005$ CNY
- **DS-R1 (671B)**: $C_{\text{real}} = 0.000333 \times 8.3 = 0.0027639 \approx 0.00276$ CNY

A.4.5 Real Deployment Cost Ratio (Qwen3-4B: Qwen3-30B: DS-R1)

$$1 : \frac{0.0005}{0.000033} : \frac{0.00276}{0.000033} \approx \mathbf{1 : 15 : 84}$$

A.5 Comparative Analysis of Two Cost Dimensions

After comparing the two cost measurement models above, we arrive at the following two conclusions:

1. **Cost Trend Consistency**:Both the market Token cost and the H200 real deployment cost show a monotonically increasing trend as the model parameter scale expands. This verifies that the Token cost model can accurately capture the core rule that "max model scale corresponds to higher cost", providing a reliable theoretical basis for the design of agent reward functions.

2. **Convergence of Cost Savings Effect**:
   Cloud Service-Based:When the agent selects Qwen3-4B to replace DS-R1-671B, the theoretical cost can be saved by 96%,Qwen3-30B to replace DS-R1-671B, the theoretical cost can be saved by 50%.
   Private Deployment-Based:When the agent selects Qwen3-4B to replace DS-R1-671B, the theoretical cost can be saved by 99%,Qwen3-30B to replace DS-R1-671B, the theoretical cost can be saved by 82%.

Model	Input Unit Price (CNY/Mtokens)	Output Unit Price (CNY/Mtokens)	Input Coefficient m_{in}	Output Coefficient m_{out}
Qwen3-4B	0.15	0.6	1	1
Qwen3-30B	2	8	13	13
DS-R1-671B	4	16	26	26

Table 8: Unit Prices and Cost Coefficients of Three Models in the Market

In terms of cost savings ratio, the savings calculated under the private deployment model are consistently higher than those under the cloud service invocation model.

From Table 9, we can see that the cost trends of the two calculation models are fully consistent. The amplified cost difference in real deployment scenarios further highlights that using an agent to select smaller models results in significantly greater actual cost savings than those calculated under the cloud service model.

B More Details of our Experiments

In this section, we provide more experimental details.

B.1 Generator Model Performance Evaluation

Tests on RAGFlag were conducted using the HotpotQA(questions Q1–Q1000) dataset for both native non-decomposed and decomposed multi-hop tasks. The primary objectives are dataset validation and model selection across small, medium, and large scales. Test results are presented in Table 10. We need a clear performance and cost gradient among max, middle, min models to create a meaningful and challenging decision space for the reinforcement learning agent. Based on our evaluation, we select **DS-R1-671B**, **Qwen3-30B**, and **Qwen3-4B** as the max, middle, and min models, respectively.

Additionally, we must determine the query decomposition model. To this end, we conduct a model selection study between DS-R1-671B and Qwen3-30B for query decomposition, with detailed results shown in Table 11. The evaluation is performed on the HotpotQA dataset (questions Q1–Q1000).

As shown in Table 11, using Qwen3-30B for question splitting yields performance comparable to DS-R1 on max models, but exhibits significant divergence on min models. To provide the agent with a more challenging and meaningful decision space, all experiments in this project adopt Qwen3-30B as the question-splitting model across all scenarios.

B.2 Detailed Construction of the Expert Rule Model Baseline

The Expert Rule Model represents a high-fidelity formalization of industry-standard heuristics, incorporating hierarchical narrowing prediction calibrated by multiple domain experts. using the same set of static and temporal features as the RL agent, ensuring a fair comparison of policy sophistication rather than feature advantage. Innovatively, we adopt a hierarchical narrowing prediction algorithm combined with human expert experience, which significantly reduces rule distortion caused by individual subjective experience and provides high-quality training data for model optimization. Specifically, the three-layer decision strategy of the rule (min/middle/max model selection) covers the full spectrum of reasoning complexity scenarios, and its parameter thresholds are calibrated via consensus from multiple domain experts rather than a single individual. By establishing this robust rule-based baseline, we eliminate the interference of feature differences and highlight the superiority of the RL agent's dynamic decision-making over static rule-based strategies in balancing reasoning accuracy and computational cost. The pseudocode for the implementation is shown in Figure 5

C More Details of Test Results

Using HotpotQA [Q1–Q1000] as the training and validation set and HotpotQA [Q1001–Q3000] as the test set, we evaluate the selected max/middle/min models, the expert rule baseline, and the agent's V1 version. The results are as follows:

On the training/validation set, RAG-on-a-Diet

Model	Token-Normalized Cost (CNY)	H200 Real Deployment Cost (CNY)	Token Cost Ratio	Real Cost Ratio
Qwen3-4B	0.00061	0.000033	1	1
Qwen3-30B	0.0079	0.0005	13	15
DS-R1-671B	0.0158	0.00276	26	84

Table 9: Cost Comparison Across Models

LLM	**Simple RAG**	**Deep Research RAG**
Qwen3-4B	0.5057	0.5657
Qwen3-30B	0.6372	0.7121
Qwen2-14B	0.4459	0.5034
Qwen2-70B	0.5163	0.5955
DS-R1-671B	0.6576	0.8045

Table 10: Generator Model Selection Test Results: DS-R1, Q3-30B, and Q3-4B exhibit the most distinct gradients in both model size and F1 score performance.

Question splitting model	**DS-R1**		**Qwen3-30B**	
	F1	**Cost**	**F1**	**Cost**
DS-R1-671B	0.7628	39.42	0.748	42.97
Qwen3-30B	0.6935	20.2	0.6415	20.2
Qwen3-4B	0.690	1.22	0.5830	1.43

Table 11: Comparison of multi-hop question splitting performance: Using Qwen3-30B for question splitting preserves the quality gradient among max, middle, and min models.

achieves a 1.6% F1 drop but a 56.08% cost reduction compared to IRCoT paired with DS-R1-671B. On the test set, it incurs a 4.46% F1 drop while reducing cost by 52.99%. These results demonstrate that, on unseen test questions (Q1001–Q3000), our trained reinforcement learning agent successfully trades less than 4.5% F1 degradation for a substantial 52.99% cost saving. This precisely quantifies the value of our framework, fulfills the project's core objective, and validates the feasibility of the "RAG-on-a-Diet" concept. Detailed results are provided in Table 12.

Based on test data analysis, the agent exhibits a 4.4% F1 drop on the test set—substantially larger than the 1.6% drop observed on the training/validation set—indicating potential overfitting. We therefore further optimize the model. The optimization strategy focuses on the following directions:

Algorithm 1 Multi-hop Reasoning Model Routing Selection

Input: Question Q
Output: Answer A

```
// Step 1: Question Complexity Classification
hop_count = CalculateQuestionHops(Q)
if hop_count ≤ 2 then short_flow_mode else long_flow_mode
// Step 2: Entity-Relation Analysis (Loopable)
repeat
  Extract entities and relations from Q
  Calculate entity importance and relation confidence
  // Step 3: Core Model Strategy Selection
  q_len = question length; avg_ent = avg entity importance
max_ent = max entity importance; prev_q = previous quality score
entity_num = entity count; bridge_flag = bridge indicator
// Layer 1: High-precision Min Strategy
if (q_len ≤6 ∧ avg_ent≥0.999 ∧ max_ent≥0.999 ∧ prev_q≤0.3 ∧ entity_num≥1) then model="min", steps=1-2
elif (q_len≤5 ∧ perfect entity importance ∧ prev_q≤0.2) then model="min", steps=1-2
elif (q_len≤4 ∧ entity importance≥0.999 ∧ prev_q=0) then model="min", steps=1-2
elif (bridge_flag exists ∧ q_len≤5 ∧ entity importance≥0.999) then model="min", steps=1-2
// Layer 2: Medium-complexity Middle Strategy
elif (6<q_len≤8 ∧ entity importance≥0.95 ∧ prev_q≤0.5) then model="middle", steps=2-3
elif (q_len>8 ∧ entity importance≥0.99 ∧ prev_q≤0.4) then model="middle", steps=3-4
elif (q_len≥10 ∧ within prev_quality/entity importance range) then model="middle", steps=2-3
elif (8.5≤q_len≤10 ∧ medium entity importance range) then model="middle", steps=2-3
// Layer 3: Full-coverage Max Strategy
else model="max", steps=3-5
// Step 4: Execute Reasoning and Generate Answer
reasoning_chain = ExecuteReasoning(model, Q, entities, relations, steps)
A = GenerateAnswer(reasoning_chain)
// Step 5: Loop Termination Check
until reasoning chain is complete and answer is accurate
return A
```

Figure 5: Multi-hop Reasoning Model Routing Selection

Interpretability and attribution analysis of agent behavior: We randomly sample instances from the validation set where the agent performs poorly (low F1 or decisions inconsistent with the expert rule model). This systematic diagnostic phase uncovered critical challenges inherent in multi-hop RL, specifically the 'first-hop bias' and the necessity for sharper reward differentiation:

- **First-hop bias:** For questions of medium length, the agent consistently prefers the middle-sized model, leading to early errors and reduced F1 scores in the first hop.
- **Suboptimal reward shaping:** The reward function treats high-quality/high-complexity and low-quality/low-complexity cases too similarly, resulting in overly flat reward signals. Consequently, the agent frequently defaults to the middle model, underperforming on high-complexity,

Methods	Gen.Model	HotpotQA (Train&Val) [Q1–Q1000]		HotpotQA (Test) [Q1001–Q3000]	
		F1	Cost	F1	Cost
IRCoT	DS-R1-671B	0.748	42.97	0.7531	41.7
	Qwen3-30B	0.6415	20.2	0.6821	15.6
	Qwen3-4B	0.583	1.43	0.6046	1.21
Expert Rule Model	Qwen3-4B/Q3-30B/DS-R1	0.7379	26.75	0.7463	29.7
RAG-on-a-DietV1	Qwen3-4B/Q3-30B/DS-R1	0.7358	18.87	0.7195	19.6

Table 12: Comparison of RAG-on-a-DietV1 with other models on HotpotQA: The V1 model achieves cost savings exceeding 50% over DS-R1 while keeping F1 drop within 5%, and reduces resource usage by over 30% compared to expert rules —– demonstrating the feasibility of our approach.

Methods	HotpotQA (T&V)		HotpotQA (Test)	
	F1	Cost	F1	Cost
V1	0.7239	19.35	0.7261	19.43
A	0.7251	23.33	0.7269	23.59
B	0.7261	16.90	0.7377	17.32
V2	0.7268	16.96	0.7387	17.35

Table 13: Comparison of optimized versions: Adjusting for first-hop bias (Version A) slightly increases cost with minimal quality gain; refining the reward function (Version B) enables better decisions; Version V2 achieves higher quality at lower cost than V1, demonstrating improved efficiency and generalization.

high-quality tasks.

Overfitting mitigation: To address potential overfitting in V1, we expand both the training and validation sets from 1,000 to 3,000 samples and evaluate the generalization performance of V1 and V2 on the test set.

To address these issues, we implement targeted improvements:

- **Version A**: Fixes the first-hop bias observed in V1;
- **Version B**: Adjusts the reward function to provide sharper differentiation;
- **Version V2**: Integrates all optimizations from A and B.

We conduct comparative evaluations of V1, A, B, and V2. Results are summarized in Table 13.

D Agent Overhead, Latency and Model Switching Overhead

We provide full quantitative details on agent overhead, latency comparison, model scale, and deployment efficiency as committed in the rebuttal.

D.1 Agent Latency and Scale Comparison

We compare our lightweight agent with base LLMs in terms of parameter count and per-hop latency.

Component	Params	Avg. Latency	Rel. Overhead
DS-R1 (671B)	671B	8.3 s/hop	100%
Qwen3-30B	30B	1.5 s/hop	18.07%
Our Agent	**150K**	**9.05 ms/hop**	**0.11%**

Table 14: Latency and scale comparison. The agent's decision time is negligible relative to LLM inference.

As shown in Table 14, our 150K-parameter CPU-based agent introduces only 9.05 ms per-hop latency, which is merely 0.11% of DS-R1's per-hop inference time on H200 hardware. This makes our agent **three orders of magnitude lighter** than routing mechanisms based on 3B-parameter LLMs such as Router-R1, which require GPU inference for every decision.

D.2 Model Switching Overhead

We eliminate all model switching overhead in our deployment pipeline:

- All models are pre-loaded in vLLM with FP8 precision on H200 GPUs.
- Routing is performed via API endpoint dispatch.
- Zero memory reallocation and no context-switch delay during inference.

D.3 3-hop Latency and Speedup

We further report end-to-end 3-hop latency and speedup across different model configurations.

Table 15 shows that in typical S+M+S scenarios, RAG-on-a-Diet achieves 1.73s 3-hop latency, corresponding to a **14.4× speedup** over the full DS-R1 baseline. Even in complex L+M+term cases, it maintains a 2.5× speedup, demonstrating strong efficiency for real-world deployment.

Configuration	3-hop Latency	Speedup
All DS-R1-671B (IRCoT)	24.9 s	1.0×
All Qwen3-30B	4.5 s	5.5×
All Qwen3-4B	0.3 s	83×
RAG-on-a-Diet (typical S+M+S)	**1.73 s**	**14.4**×
RAG-on-a-Diet (complex L+M+term)	**9.82 s**	**2.5**×

Table 15: 3-hop inference latency and speedup across different configurations.

Dropout p	Test F1	Test Cost	ΔF1 vs. full model
0 (full)	0.7387	17.35	—
0.1	0.7354	17.82	-0.45%
0.2	0.7311	18.41	-1.03%
0.3	0.7248	19.15	-1.88%

Table 16: Feature dropout noise robustness analysis. ΔF1 denotes the F1 score drop relative to the full model ($p = 0$).

E Noise Robustness via Feature Dropout

To analyze the noise robustness of our static and temporal features, we perform feature dropout experiments by randomly masking input features with probabilities $p = 0.1, 0.2, 0.3$. Table 16 summarizes the F1 score and inference cost under different settings.

The results demonstrate that our model maintains stable performance even under moderate feature noise. Even at $p = 0.3$, the F1 drop is marginal and cost remains well-controlled, confirming the robustness and redundancy of our feature design.

F More Details of Ablation Test Results

F.1 Conclusions of Static Feature Ablation Tests

In this experiment, single-feature ablation tests were conducted for four dimensions of static features, namely entity count, question length, entity importance, and relationship salience. The version with entity count ablated is denoted as Diet-Abl-NE; the version with question length ablated is denoted as Diet-Abl-QL; the version with entity importance ablated is denoted as Diet-Abl-AEI; and the version with relationship salience ablated is denoted as Diet-Abl-ARC. The comparison results of these tests on the HotpotQA dataset are presented in Table 17.

From the F1 and cost metrics on the training&validation and test sets, all four static features significantly influence the model's quality–cost balance, with varying degrees of importance and distinct functional roles. Based on the magnitude of F1 degradation and cost change, we rank the features by their impact and direction as follows:

Methods	Training&Validation Set HotpotQA		Test Set HotpotQA	
	F1	Cost	F1	Cost
RAG-on-a-DietV2	0.7268	16.96	0.7387	17.35
Diet-Abl-NE	0.7195	22.67	0.7325	22.1
Diet-Abl-QL	0.7015	15.15	0.72	15.31
Diet-Abl-AEI	0.7113	20.24	0.726	20.73
Diet-Abl-ARC	0.7143	21.29	0.734	21.89

Table 17: Static feature ablation test results: Four single-feature ablation experiments show that each feature is strongly correlated with quality, cost, and the quality-cost balance, and makes a practical contribution to the agent.

- **Entity Count**: Plays the most critical role in cost control, with a mild positive effect on quality.
- **Entity Importance**: Strongly affects both quality (significant drop) and cost (substantial increase), making it a key quality–cost coupling feature.
- **Relation Salience**: Induces moderate F1 loss but significant cost growth, indicating strong redundancy in cost allocation.
- **Question Length**: Significantly impacts quality while offering cost savings —— a "weakly quality-related, strongly cost-redundant" feature.

Analyzing the action distribution after ablation, we observe that ablating entity count leads to a 6.5% increase in "max" actions compared to Version V2 —— this is the primary reason for the significant cost rise after ablation. Detailed data can be found in Figure 6. Further fine-grained feature analysis yields consistent conclusions. For instance, examining the distributions along the question length and entity count dimensions reveals that, in each category, the ablated versions consistently exhibit higher proportions of "max" actions than V2 (see Figures 7 and 8). This confirms that entity count is strongly correlated with cost expenditure.
After ablating question length, the ablated version exhibits a significant increase in "min" actions—— by nearly 25%. This pattern holds consistently across all feature dimensions: in every breakdown (including queries involving high-importance entities and complex questions), the ablated model

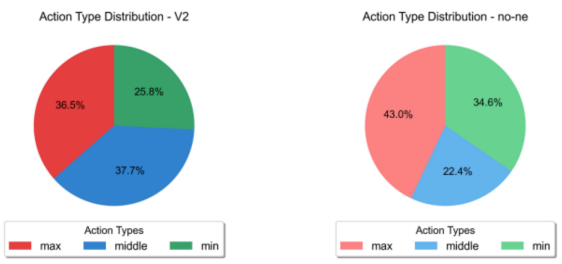

Figure 6: Comparison of action distributions between RAG-on-a-DietV2 and Diet-Abl-NE: after ablating entity count, the agent uses fewer "middle" actions and exhibits a more aggressive policy overall in handling questions.

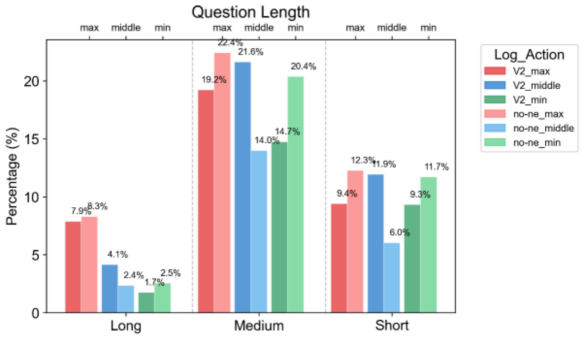

Figure 7: Comparison of action distributions between RAG-on-a-DietV2 and Diet-Abl-NE: after ablating entity count, the agent uses fewer "middle" actions and exhibits a more aggressive policy overall in handling questions.

selects "min" actions substantially more often than V2. This overuse of the min model directly contributes to the drop in F1 score, confirming that question length is a strong quality–cost coupling feature.

F.2 Conclusions of Temporal Feature Ablation Tests

In this ablation experiment, single-feature ablation tests were conducted for two dimensions of temporal features——global quality and global salience——based on the RAG-on-a-Diet version. The version with global quality ablated is denoted as Diet-Abl-CQ, and the version with global salience ablated is denoted as Diet-Abl-ORC. Detailed comparative tests were also performed on the HotpotQA dataset, and the results are presented in Table 18.

From the F1 and cost results on both training/validation and test sets, the two temporal features significantly influence the quality–cost tradeoff, with distinct roles:

- **Global Quality** is crucial for **cost control** and mildly improves quality. Ablation (Figure 9) shifts the action distribution: "max" and "min" increase, while "middle" decreases, leading to higher cost and degraded training-set F1 (test F1 only slightly drops). Figures 10 and 11 show that after ablation, high-quality or complex subqueries are over-allocated to "max" or under-allocated to "min", causing resource misallocation. Notably, performance does not collapse because *current-hop quality* —— which incorporates information from both the current and previous hop —— provides redundancy, revealing the agent's robustness.

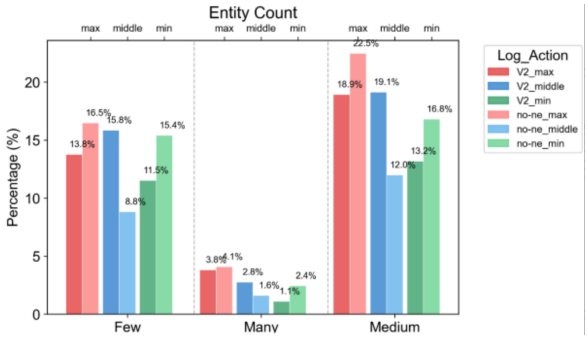

Figure 8: Comparison of action distributions w.r.t. the entity count feature between RAG-on-a-Diet V2 and Diet-Abl-NE: after ablating the entity count feature, the statistics along the entity count dimension clearly confirm that entity count is strongly correlated with cost.

- **Global Confidence** plays a **more critical role in quality (F1)** and offers moderate cost benefits, though less than global quality. Ablation (Figure 12) also reduces "middle" actions and increases both "max" and "min", but with *larger "min" growth* and *smaller "max" growth* than global quality ablation —— resulting in a clearer F1 drop. Figure 13 shows this misallocation is especially severe on complex/medium questions (more "min", less "middle") and simple questions (more "max" and "min"). Figure 14 confirms that even for high-quality previous hops, the ablated agent overuses extreme actions, validating the importance of global confidence for balanced decision-making.

F.3 Conclusions of Feature Combination Ablation Experiments

In this ablation experiment, combined ablation tests were performed on static features and temporal features. The version with the combination of entity count and entity importance ablated is denoted as

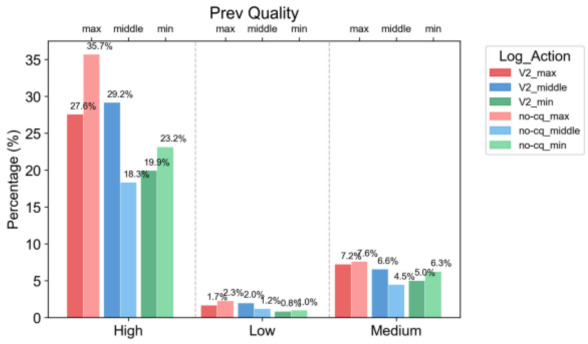

Figure 9: Comparison of action distributions between RAG-on-a-Diet V2 and Diet-Abl-CQ: without access to global quality as a reference, the agent assigns most subtasks—originally handled by "middle"—to the "max" model, leading to significantly higher overall cost.

Figure 10: Comparison of action distributions w.r.t. previous-hop quality between RAG-on-a-Diet V2 and Diet-Abl-CQ: after ablating global quality, the agent relies solely on current-hop and previous-hop quality as its quality signal, leading to overly conservative decisions. Specifically, within the high-quality previous-hop category, the ablated agent significantly increases "max" actions while sharply reducing "middle" usage—resulting in higher overall cost.

Diet-Abl-NEAEI; the version with the combination of question length and relationship salience ablated is denoted as Diet-Abl-QLARC; the version with the combination of global quality and global salience ablated is denoted as Diet-Abl-CQORC; the version with the combination of global quality, previous-hop quality and global salience ablated is denoted as Diet-Abl-CQPQORC; and the version with the combination of entity count, entity importance, global quality and previous-hop quality ablated is denoted as Diet-Abl-NEAEICQPQ. Detailed comparative tests were also conducted using the HotpotQA dataset for these ablation versions, and the results are presented in Table 19.

After ablating both entity count and entity importance, the agent's action distribution shifts significantly toward "min". Detailed analysis across feature categories reveals that this ablation removes critical reward signals, causing the agent to overselect the "min" model, which ultimately degrades

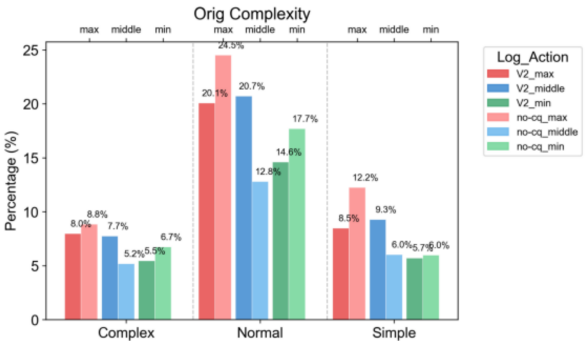

Figure 11: Comparison of action distributions w.r.t. question complexity between RAG-on-a-Diet V2 and Diet-Abl-CQ: after ablating global quality, the agent still exhibits differentiated—often aggressive—decision behavior across complexity levels. This demonstrates the compensatory role of current-hop and previous-hop quality features, which partially offset the loss of global quality and help maintain adaptive routing.

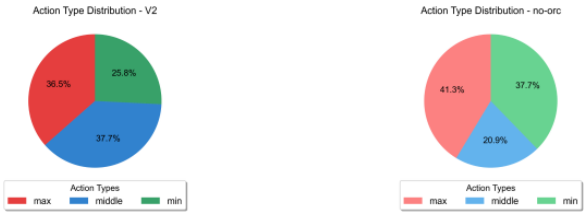

Figure 12: Comparison of action distributions between RAG-on-a-Diet V2 and Diet-Abl-CQ: after ablating global confidence, the agent also behaves more aggressively.

task quality.From Figure 15, we observe that both "max" and "middle" actions decrease, while "min" rises significantly. As shown in Table 19, this ablation leads to a **simultaneous drop in both quality and cost**, indicating that the altered reward signal causes the agent to **over-rely on the "min" model**.

Analyzing along the **question length** and **question complexity** dimensions further clarifies this behavior:

- Along **question length**, the agent shows some degree of policy differentiation—particularly exhibiting excessive trust in "min" for medium-length questions.

- However, across all categories of **question complexity**, the agent uniformly favors "min" actions, as shown in Figures 16 and 17.

This consistent bias toward "min" across complexity levels directly explains the observed degradation in both F1 and cost efficiency.

After ablating *question length* and *relation*

Methods	Training&Validation Set HotpotQA		Test Set HotpotQA	
	F1	Cost	F1	Cost
RAG-on-a-DietV2	0.7268	16.96	0.7387	17.35
Diet-Abl-CQ	0.7145	22.17	0.7356	22.51
Diet-Abl-ORC	0.7093	19.49	0.7207	20.62

Table 18: Temporal feature ablation test results: These features are clearly strongly correlated with cost—cost rises sharply after ablation. After ablating global quality, the agent still receives reward signals from current quality (including previous-hop quality), so there is little deviation in quality from the test results. However, after ablating global confidence, quality drops significantly: when the agent loses the ability to judge question reliability, the system overhead is reduced via the reward of the quality-cost balance coefficient.

Methods	Training&Validation Set HotpotQA		Test Set HotpotQA	
	F1	Cost	F1	Cost
RAG-on-a-DietV2	0.7268	16.96	0.7387	17.35
Diet-Abl-NEAEI	0.6906	15.33	0.7088	16.38
Diet-Abl-QLARC	0.6745	10.01	0.6991	10.57
Diet-Abl-CQORC	0.7112	20.22	0.7240	21.37
Diet-Abl-CQPQORC	0.7102	16.12	0.7198	16.81
Diet-Abl-NEAEICQPQ	0.7059	17.88	0.7171	18.34

Table 19: Combined feature ablation test results: For the HotpotQA dataset, the combinations of "entity count + entity importance" and "question length + relation confidence" have a significant impact on the F1 score; the ablation of "global quality + global confidence" significantly affects cost. After ablating "global quality + previous-hop quality + global confidence", with only the current-hop quality serving as the agent's reference, the system takes actions that reduce both quality and cost based on the quality-cost balance.

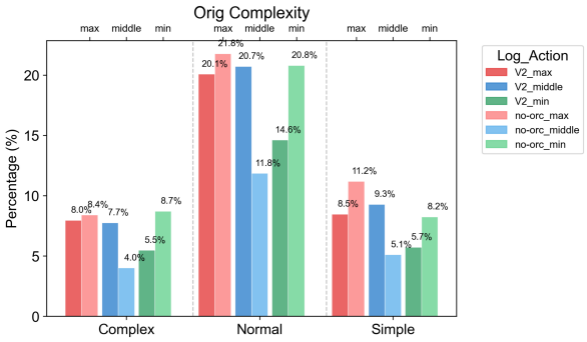

Figure 13: Comparison of action distributions w.r.t. question complexity between RAG-on-a-Diet V2 and Diet-Abl-ORC: after ablating global confidence, the agent adopts a more aggressive policy—significantly increasing "min" actions on complex questions and "max" actions on simple ones. This mismatch in resource allocation poses challenges to both cost efficiency and answer quality.

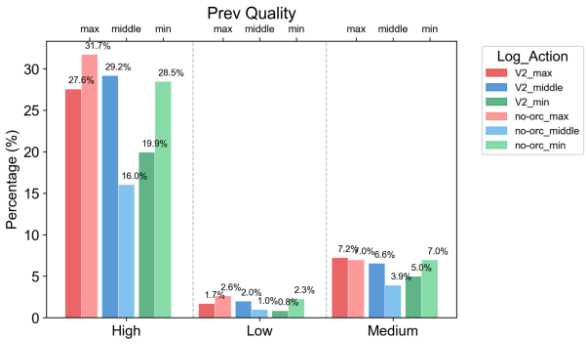

Figure 14: Comparison of action distributions w.r.t. previous-hop quality between RAG-on-a-Diet V2 and Diet-Abl-ORC: the trend aligns with the overall shift in action distribution observed across the entire dataset.

salience, the agent essentially defaults to the "min" action indiscriminately. Across all feature-based subgroups, the action distribution shows a consistent and overwhelming preference for "min". This indicates that the agent has lost most of its ability to characterize input questions and match them to the appropriate model. See Figures 18 and 19 for details.

After ablating both *global quality* and *global confidence*, the agent exhibits a *dual degradation* in both quality and cost. The action distribution (compared to V2) becomes overly aggressive: "middle" actions decrease, while both "max" and "min" increase. Nevertheless, thanks to reward signals from other features, the agent avoids catastrophic quality collapse. Feature-wise analysis shows the agent retains some policy differentiation; however, when stratified by question complexity, it becomes overly

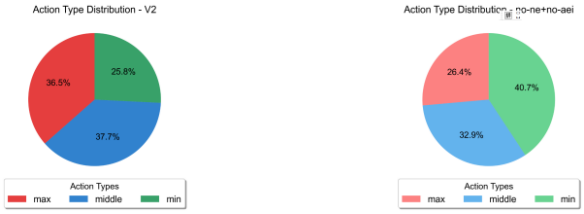

Figure 15: Comparison of action distributions between RAG-on-a-Diet V2 and Diet-Abl-NEAEI: after ablating both entity count and entity importance, the system exhibits a simultaneous decline in both overall quality and cost, indicating that the agent has become overly reliant on the "min" action.

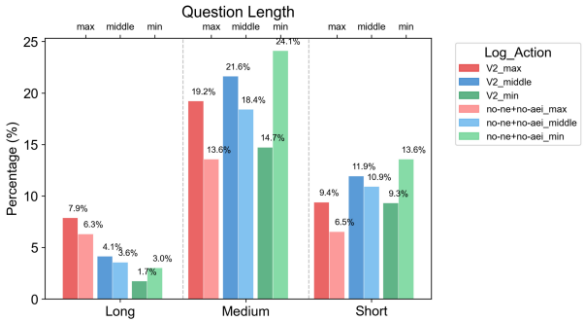

Figure 16: Comparison of state-action distributions w.r.t. question length between RAG-on-a-Diet V2 and Diet-Abl-NEAEI: after ablating entity count and entity importance, the agent retains some policy differentiation across question lengths, but exhibits a clear over-reliance on the "min" action for medium-length questions.

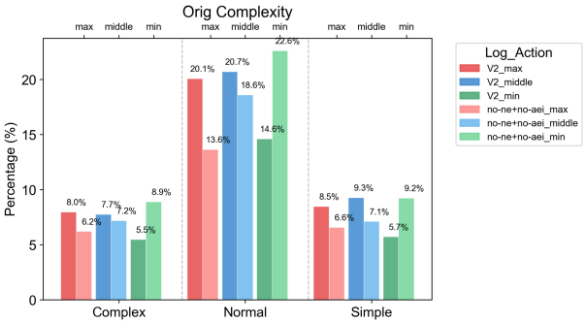

Figure 17: Comparison of state-action distributions w.r.t. question complexity between RAG-on-a-Diet V2 and Diet-Abl-NEAEI: after ablating entity count and entity importance, the agent exhibits nearly homogeneous behavior across all complexity categories. This indicates that this feature combination is crucial for the agent to assess question complexity and select the optimal action accordingly.

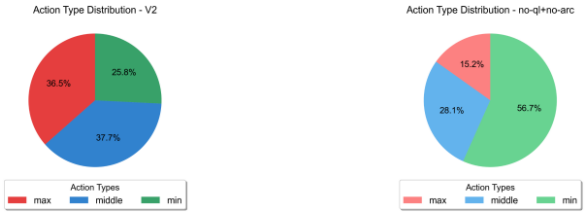

Figure 18: Comparison of action distributions between RAG-on-a-Diet V2 and Diet-Abl-QLARC: after ablating question length and relation salience, the agent's ability to discern question characteristics weakens substantially, causing the reward signal to overwhelmingly bias the policy toward the "min" action.

conservative on simple questions and excessively aggressive on complex ones. See Figures 20 and 21 for empirical evidence.

After ablating global quality, global confidence, and previous-hop quality, the agent's action distribution shifts strongly toward "min". By removing previous-hop quality in addition to the other two, all quality signals are reduced to the current-hop quality alone. Consequently, the agent's decisions become dominated by cost-sensitive reward signals, leading to a consistent preference for resource-saving actions across all feature categories—manifested as a pronounced increase in "min" selections. See Figures 22 and 23 for detailed results.

In the combined-feature ablation studies, feature pairs such as *entity count & entity importance* and *question length & relation salience* are critical for the model to perceive task complexity and select appropriate actions. Removing these features severely degrades the agent's ability to make targeted and appropriately aggressive decisions: in the QL-ARC ablation, the agent blindly defaults to "min", causing significant quality degradation; in the CQ-ORC ablation, action selection becomes erratic, leading to concurrent drops in both performance and cost efficiency. These results validate the importance of the *aggressive reward-shaping strategy* in our reward function, which explicitly leverages these key features to guide effective decision-making.

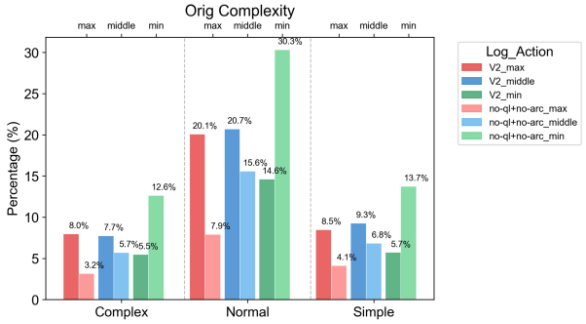

Figure 19: Comparison of state-action distributions w.r.t. question complexity between RAG-on-a-Diet V2 and Diet-Abl-QLARC: after ablating question length and relation salience, the agent exhibits an overwhelming dominance of the "min" action across nearly all feature categories, effectively losing its ability to differentiate strategies based on question characteristics.

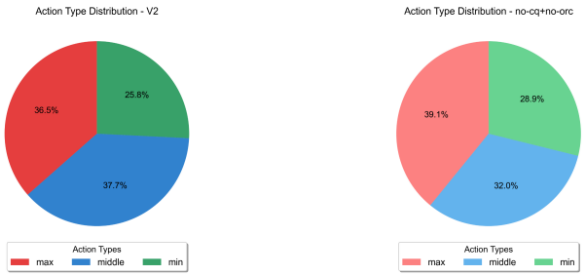

Figure 20: Comparison of action distributions between RAG-on-a-Diet V2 and Diet-Abl-CQORC: after ablating global quality and global confidence, the agent adopts a more aggressive policy, with a notable decrease in "middle" actions.

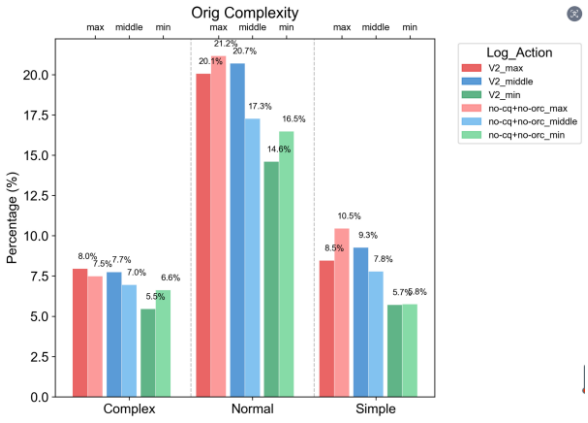

Figure 21: Comparison of state-action distributions w.r.t. question complexity between RAG-on-a-Diet V2 and Diet-Abl-CQORC: the agent exhibits differentiated behavior across complexity levels, but the reward signal appears overly conservative for *simple* questions and excessively aggressive for *complex* ones.

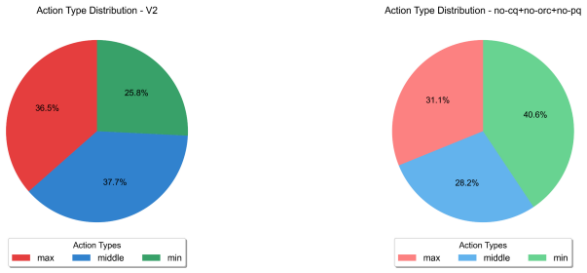

Figure 22: Comparison of action distributions between RAG-on-a-Diet V2 and Diet-Abl-CQPQORC: after ablation, the agent clearly shifts toward the "min" action.

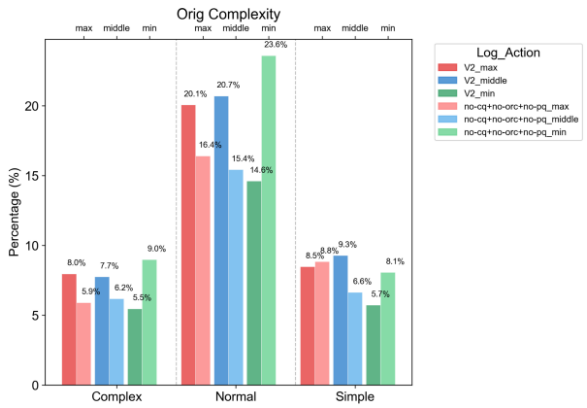

Figure 23: Comparison of question complexity feature distributions between RAG-on-a-Diet V2 and Diet-Abl-CQPQORC: across nearly all feature-based state categories, the agent exhibits a significant increase in "min" actions and a marked decrease in both "max" and "middle" actions, indicating that it has effectively lost access to quality-preserving reward signals.