TriCo-RAG: Tri-Stage Adaptive Coordination for Multi-Source Retrieval-Augmented Generation

Anonymous ACL submission

Abstract

Retrieval-augmented generation (RAG) has been widely adopted to mitigate hallucination in large language models (LLMs) by grounding model outputs in external evidence. Recent work explores multi-source retrieval to obtain more comprehensive evidence. However, existing approaches typically lack adaptive coordination in multi-source RAG, including (1) adapting retrieval source selection to the information needs of different queries, (2) adapting queries to better utilize different retrieval sources, and (3) selecting evidence under fixed token budgets using query-side cues. To address these challenges, we propose TriCo-RAG, a tri-stage adaptive coordination framework for multi-source retrieval. TriCo-RAG jointly coordinates retrieval source selection, query adaptation for different retrieval sources, and evidence utilization. Experiments on multiple QA benchmarks show that TriCo-RAG consistently improves answer accuracy while using fewer evidence tokens, with stable gains across model scales from Llama 3.1-8B to Qwen3-Max and GPT-5.

1 Introduction

Large language models (LLMs) have demonstrated strong reasoning and generation capabilities (Wei et al., 2022; Touvron et al., 2023; OpenAI, 2024; Llama Team, 2024). To improve factual grounding, retrieval-augmented generation (RAG) augments model generation with externally retrieved evidence (Lewis et al., 2020).

Early RAG systems rely on a single retrieval source (e.g., a static corpus), which may provide limited evidence diversity for complex queries (Izacard and Grave, 2021; Shi et al., 2023); Fig. 1(A) illustrates this setting. Recent work accordingly explores multi-source retrieval by combining different retrieval sources, such as structured knowledge access and web retrieval (Agrawal et al., 2025; Tan et al., 2025). However, existing approaches often rely on simple coordination strategies, such as fixed source weighting (Hwang et al., 2025) or coarse tool routing (Yao et al., 2023; Guo et al., 2026). As illustrated in Fig. 1(B), existing multi-source RAG approaches often lack a holistic adaptive mechanism for coordinating retrieval sources and evidence utilization, which hinders effective use of multi-source evidence (Li et al., 2025).

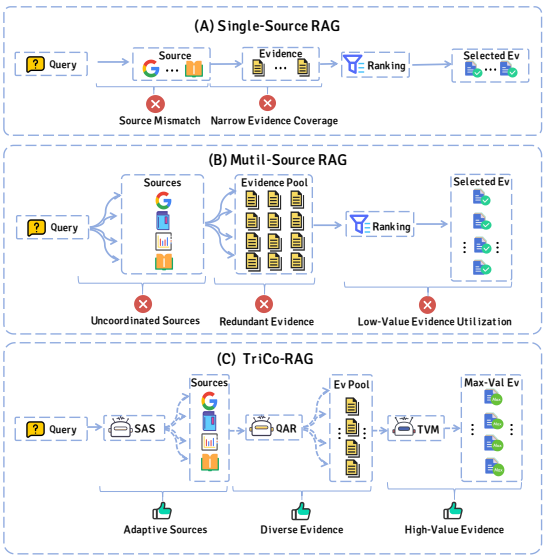


Figure 1: **Comparison of retrieval paradigms in RAG. (A)** Single-source RAG with a fixed retrieval source. **(B)** Naive multi-source RAG with pooled evidence but without coordinated adaptation. **(C)** TriCo-RAG with tri-stage coordination via SAS, QAR, and TVM.

We argue that adaptive multi-source retrieval introduces three key challenges.

(1) Query-guided retrieval source selection. Existing routing and source selection strategies often rely on coarse heuristics that may not accurately reflect a query's information needs (Yao et al., 2023; Hwang et al., 2025; Guo et al., 2026). Since retrieval sources specialize in different types of evidence (Jiang et al., 2023), suboptimal source selection may retrieve evidence that only partially

addresses the query and degrades downstream generation quality (Izacard and Grave, 2021; Shi et al., 2023).

(2) Query adaptation across retrieval sources. The original query alone may not fit how different retrieval sources retrieve evidence (Ammann et al., 2025). As a result, retrieval may capture partial or overlapping evidence across sources. Existing reformulation strategies often lack explicit source-aware query adaptation for multi-source retrieval (Li et al., 2025; You et al., 2025).

(3) Budget-aware evidence utilization. Multi-source retrieval enlarges the candidate evidence pool (Agrawal et al., 2025; Tan et al., 2025) but also introduces redundancy and higher token cost (Liu et al., 2024). Under fixed token budgets, many pipelines still truncate or rank evidence using retrieval scores or passage length alone (Cheng et al., 2024; Panda et al., 2025), without incorporating query-side retrieval cues from upstream reformulation. As the candidate pool grows, this decoupling makes it harder to balance relevance, complementary coverage, and token efficiency.

To address these challenges, we propose **TriCo-RAG**, a tri-stage adaptive coordination framework for multi-source retrieval with three mechanisms corresponding to challenges (1)–(3).

(1) **Source-Adaptive Selection (SAS)** coordinates retrieval sources according to a query's information needs and source specializations. Unlike fixed hybrid pipelines (Tan et al., 2025) or prompt-based tool routing (Yao et al., 2023), SAS dynamically selects retrieval sources for each query.

(2) **Query-Adaptive Reformulation (QAR)** generates diversified query variants to better exploit different retrieval sources. Instead of relying only on the original query or explicit decomposition (Ammann et al., 2025), QAR constructs query variants with different semantic cues for different retrieval sources, thereby improving evidence diversity in multi-source retrieval (You et al., 2025).

(3) **Token-Value Maximization (TVM)** selects evidence under a fixed token budget. Unlike score-heavy truncation, length-first packing, or rerank-only value models that do not consume QAR cues (Cheng et al., 2024; Wang et al., 2025), TVM greedily maximizes *newly covered reformulation cues per token*, coupling budgeted selection with upstream query adaptation.

Together, these mechanisms enable adaptive coordination for more effective evidence utilization in multi-source retrieval, as illustrated in Fig. 1(C).

Experiments on HotpotQA (Yang et al., 2018), TriviaQA (Joshi et al., 2017), and MuSiQue (Trivedi et al., 2022) show that TriCo-RAG improves answer accuracy while using evidence tokens more efficiently. These gains hold across model scales, from Llama 3.1-8B (Llama Team, 2024) to Qwen3-Max[1] and GPT-5[2].

We summarize our contributions as follows:

- **A formulation for adaptive multi-source RAG.** We formulate multi-source RAG as a coordinated evidence utilization problem and propose TriCo-RAG, a unified framework that jointly coordinates adaptive retrieval and evidence utilization.
- **Three modular mechanisms for adaptive multi-source retrieval.** We design Source-Adaptive Selection (SAS), Query-Adaptive Reformulation (QAR), and Token-Value Maximization (TVM) to address query-guided source selection, cross-source query adaptation, and budget-aware evidence utilization, respectively. These mechanisms are modular and can be integrated into different retrieval settings.
- **Efficient evidence utilization under token budgets.** Experiments show TriCo-RAG improves answer accuracy while reducing evidence token usage. Further analyses demonstrate the effectiveness of cue-aligned, token-aware evidence selection under constrained budgets.

2 Related Work

Retrieval-augmented generation (RAG) grounds generation in externally retrieved knowledge (Guu et al., 2020; Lewis et al., 2020). Dense and hybrid retrievers (Karpukhin et al., 2020; Khattab and Zaharia, 2020; Santhanam et al., 2022; Izacard et al., 2023) as well as graph- and vector-based retrieval methods (Yu et al., 2025) have improved retrieval quality and evidence access. However, many systems still rely on fixed or manually tuned configurations, while adaptive coordination in multi-source retrieval remains underexplored.

Query-guided retrieval source selection. Recent work adapts retrieval to the query through LLM-guided routing and reasoning (Jiang et al., 2023; Yao et al., 2023; Dong et al., 2025; Li et al., 2025; Guo et al., 2026), modular retrieval architectures (Zhang et al., 2025), multi-agent coordination (Chang et al., 2025), RL-based joint optimization (Chen et al., 2025), and multi-retriever fusion (Li

[1] https://help.aliyun.com/zh/model-studio/

[2] https://developers.openai.com/api/docs/models

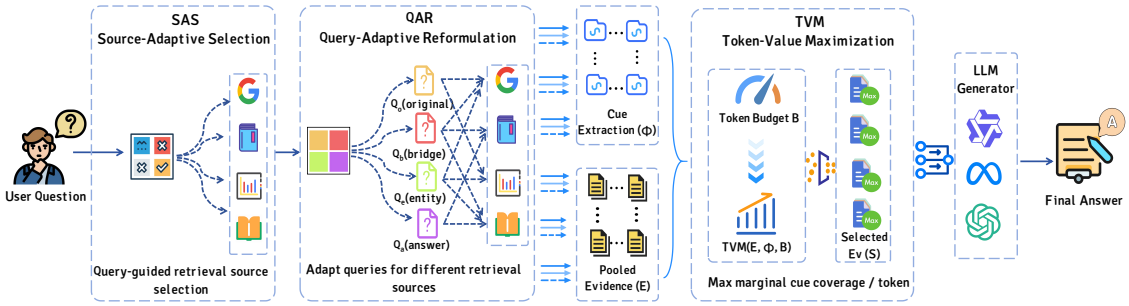


Figure 2: **Overview of the TriCo-RAG pipeline.** SAS routes each query to suitable retrieval sources. QAR adapts queries for those sources, extracts cue set Φ, and pools retrieved evidence into E. TVM selects $S \subseteq E$ by maximizing marginal cue coverage per token under budget B before generation.

et al., 2024). These methods mainly focus on retrieval routing and coordination, with relatively little attention to coordinated query adaptation across retrieval sources.

Query adaptation for retrieval diversification. Query adaptation improves retrieval recall for complex and multi-hop questions, where relevant evidence is often distributed across sources (Yang et al., 2018; Trivedi et al., 2022; Wang et al., 2023). Prior work uses interleaved reasoning and retrieval (Trivedi et al., 2023), self-knowledge-guided reformulation (Wang et al., 2023), and question decomposition (Ammann et al., 2025). However, they mainly target general retrieval settings rather than per-source query diversification.

Utility-aware evidence selection. Evidence selection is crucial for RAG systems: increasing context length does not always yield better performance (Liu et al., 2024), and effective utilization often depends on selecting high-value evidence rather than adding more tokens (Joren et al., 2025). Prior work explores value-aware selection (Cheng et al., 2024; Wang et al., 2025), multi-semantic representations (You et al., 2025), reliability-aware retrieval (Hwang et al., 2025), hierarchical or graph-based organization (Sarthi et al., 2024; Han et al., 2025), critique- and control-based refinement (Zhou et al., 2024; Sun et al., 2025; Wei et al., 2025), and alternative selection paradigms (Saxena et al., 2025). These approaches improve evidence selection under limited context windows, but upstream retrieval adaptation and downstream evidence utilization are often optimized separately.

3 Methodology

We present TriCo-RAG, a framework for tri-stage adaptive coordination in multi-source retrieval-augmented generation. For a query q, it coordinates (i) retrieval source selection, (ii) query adaptation, and (iii) cue-aligned evidence selection under a fixed token budget.

As illustrated in Fig. 2, TriCo-RAG coordinates three components for adaptive multi-source RAG. Source-Adaptive Selection (SAS) routes each query to suitable retrieval sources; Query-Adaptive Reformulation (QAR) constructs source-aware query signals that expose complementary retrieval cues; and Token-Value Maximization (TVM) uses these cues to prioritize evidence with higher cue coverage per token before generation.

3.1 Overview

Given q, TriCo-RAG applies SAS to choose one or more retrieval sources $r \in \mathcal{R}$, where $\mathcal{R}$ contains a static corpus index, a live web index, and a hybrid joint-retrieval option (concrete routes are defined in §3.2). QAR then constructs query signals $Q(q) = \{Q_o, Q_b, Q_e, Q_a\}$, where bridge query Q_b is activated only when a bridge trigger fires. These signals are fused into a unified retrieval representation over the selected source(s), from which TriCo-RAG extracts cue set Φ and retrieves pooled evidence E. TVM selects $S \subseteq E$ using Φ by maximizing marginal cue coverage per token under budget B, and passes S to the generator.

This extends the standard retrieve-then-generate paradigm (Guu et al., 2020; Lewis et al., 2020) by explicitly coupling source selection, query adaptation, and evidence selection.

3.2 Source-Adaptive Selection (SAS)

SAS performs retrieval source selection by scoring routes $r \in \mathcal{R}$ conditioned on the query. We instantiate $\mathcal{R} = \{\mathrm{CR}, \mathrm{WR}, \mathrm{CR\&WR}\}$ for corpus retrieval (CR), web retrieval (WR), and explicit joint corpus+web retrieval (CR&WR); CR&WR is a route label in three-way routing rather than a post-hoc fusion fallback.

Following adaptive retrieval and LLM-guided

routing (Jiang et al., 2023; Yao et al., 2023; Guo et al., 2026), each query is summarized as $f = \{h, d, c, t\}$: reasoning hops h, retrieval difficulty d, evidence count c, and answer type t.

Feature design. We use (h, d, c) to model routing behavior and t for answer-oriented cues: h estimates reasoning depth; d estimates retrieval difficulty under corpus retrieval; c estimates evidence count; and t represents the expected answer format (e.g., person or date). The tuple (h, d, c) is used for route-prior estimation, while t is used only by QAR to construct Q_a, as answer format mainly affects query reformulation rather than source selection. Features are inferred from q using an LLM-based extractor with temperature 0.0 (Appendix H.2).

Feature-based prior. We combine a lightweight structural prior with LLM routing because pure LLM tool selection can over-trigger web retrieval on shallow multi-hop questions that corpus evidence already covers, while a discrete prior over (h, d, c) yields stable, auditable biases. We define

$$s_{\text{prior}}(r \mid f) = \text{TablePrior}(r; h, d, c),$$

where $\text{TablePrior}(r; h, d, c) \in [0, 1]$ for all $r \in \mathcal{R}$, and $\sum_{r \in \mathcal{R}} \text{TablePrior}(r; h, d, c) = 1$.

The table is hand-specified on the development set to encode domain preferences (e.g., higher CR mass for easy 2-hop queries, higher CR&WR mass for deep multi-hop queries); full schema and tables are in Appendix H.2.

LLM-based preference. To capture signals beyond discrete features, we estimate $s_{\text{llm}}(r \mid q)$ with an LLM that scores suitable source(s) for q, following LLM-guided retrieval (Jiang et al., 2023).

Mixture and selection. The route score is

$$s(r) = \lambda\, s_{\text{prior}}(r \mid f) + (1 - \lambda)\, s_{\text{llm}}(r \mid q),$$

where $\lambda \in [0, 1]$ is tuned on a development set (λ=0.7 in our runs).

We use margin-based selection with threshold $\delta > 0$. Let r_1 and r_2 be the top two routes by $s(r)$. If $s(r_1) - s(r_2) \geq \delta$, we set $r^\star = r_1$; otherwise we keep the top-M routes with $M = 2$ to support source fusion.

3.3 Query-Adaptive Reformulation (QAR)

QAR constructs source-aware query signals to adapt query semantics to different retrieval sources.

Query-signal design. Given q, we construct four complementary signals: (i) Q_o preserves the original semantic intent; (ii) Q_b (optional) targets intermediate entities in multi-hop questions; (iii) Q_e improves entity-centric retrieval; and (iv) Q_a biases retrieval toward the expected answer format. Together, these signals provide complementary retrieval cues beyond a single query formulation (Ammann et al., 2025; You et al., 2025) and define the cue set Φ used by TVM (§3.4), linking query reformulation to budgeted evidence selection. Q_b is activated only when $\text{BridgeTrigger}(f)$ fires (Appendix H.1.2), and active signals are fused before retrieval on SAS-selected sources (Appendix H.2.2). Hybrid dense–sparse scoring is

$$s(e, q) = w_{\text{dense}}\, s_{\text{dense}}(e, q) + w_{\text{bm25}}\, s_{\text{bm25}}(e, q),$$

where e is a candidate passage, s_{dense} and s_{bm25} denote dense and BM25 scores, and w_{dense}, w_{bm25} are mixture weights (Karpukhin et al., 2020; Khattab and Zaharia, 2020; Santhanam et al., 2022). Retrieved evidence from active signals and selected sources is merged into a pooled evidence set E, with structural deduplication by canonical passage or chunk identifiers (Appendix H.2.5).

3.4 Token-Value Maximization (TVM): Evidence Selection

TVM performs cue-aligned evidence selection using retrieval cues from QAR. When multi-source pooling enlarges E, ranking by retrieval score alone often keeps redundant passages that repeat the same entities while missing bridge or answer-type cues needed for multi-hop QA. TVM instead asks: under budget B, which passages add the most *new* coverage of QAR-derived cues per token? This objective encourages complementary evidence (penalizing duplicates that cover the same ϕ) and explicitly couples selection with upstream reformulation, unlike score-only truncation or rerank-only value models that do not use QAR-derived cues (Cheng et al., 2024; Wang et al., 2025).

Cue set and coverage. From active QAR signals and lightweight extractors over q, we construct a cue set Φ (keywords, entities, bridge phrases, and answer-type phrases; Appendix H.2.5). For evidence e and cue ϕ, coverage $\text{cov}(e, \phi) \in [0, 1]$ is computed using embeddings and lexical cues. Algorithm 1 presents the corresponding thresholded view with $\text{cov}(e, \phi) \in \{0, 1\}$ for simplicity.

Objective. We define

$$\max_{S \subseteq E} f(S) \quad \text{s.t.} \quad \sum_{e \in S} \text{tokens}(e) \leq B.$$

We instantiate

$$f(S) = \left|\{\phi \in \Phi : \exists e \in S,\ \text{cov}(e, \phi) = 1\}\right|.$$

(or its soft coverage sum in implementation).

Maximizing $f(S)$ favors evidence that jointly covers reformulation cues for complex questions rather than relying only on rerank scores. The corresponding greedy selection procedure is described in Appendix H.2.5.

Optimization. We adopt a greedy strategy over

$$\mathcal{C} = \{e \in E \setminus S : \text{tokens}(e) \leq B_{\text{rem}}\}.$$

In each iteration,

$$e^\star = \arg\max_{e \in \mathcal{C}} \frac{\Delta(e)}{\text{tokens}(e)},$$

where $\Delta(e)$ is marginal *soft* cue coverage from e, plus a small rerank-score prior (Appendix H.2.5).

We warm-start with the top-k reranked passages, then greedily add items until the budget is exhausted or marginal gain falls below a threshold. The selected set S is then passed to the generator for answer generation (Lewis et al., 2020). Algorithm 1 summarizes TVM; Appendices H.1.1 and H.2.5 give SAS+QAR pseudocode, token accounting, and implementation details.

Algorithm 1: Token-Value Maximization

Input: ranked evidence set E; token budget B; cue set Φ; warm-start size k; coverage $\text{cov}(e, \phi) \in [0, 1]$ (thresholded in lines below)
Output: selected evidence set S

$S \leftarrow \emptyset$, ;
$B_{\text{rem}} \leftarrow B$, ;
$\Phi_{\text{cov}} \leftarrow \emptyset$;
for *e in top-k items of E (rank order)* **do**
  if $\text{tokens}(e) \leq B_{rem}$ **then**
    $S \leftarrow S \cup \{e\}$;
    $B_{\text{rem}} \leftarrow B_{\text{rem}} - \text{tokens}(e)$;
    $\Phi_{\text{cov}} \leftarrow \Phi_{\text{cov}} \cup \{\phi \in \Phi : \text{cov}(e, \phi) = 1\}$;
while $B_{rem} > 0$ **do**
  $\mathcal{C} \leftarrow \{e \in E \setminus S : \text{tokens}(e) \leq B_{\text{rem}}\}$;
  if $\mathcal{C} = \emptyset$ **then**
    break;
  for $e \in \mathcal{C}$ **do**
    $\Delta(e) \leftarrow \left|\{\phi \in \Phi : \phi \notin \Phi_{\text{cov}} \wedge \text{cov}(e, \phi) = 1\}\right|$;
  $e^\star \leftarrow \arg\max_{e \in \mathcal{C}} \Delta(e)/\text{tokens}(e)$;
  if $\Delta(e^\star) = 0$ **then**
    break;
  $S \leftarrow S \cup \{e^\star\}$,;
  $B_{\text{rem}} \leftarrow B_{\text{rem}} - \text{tokens}(e^\star)$;
  $\Phi_{\text{cov}} \leftarrow \Phi_{\text{cov}} \cup \{\phi \in \Phi : \text{cov}(e^\star, \phi) = 1\}$;
return S;

4 Experimental Setup

4.1 Datasets and Metrics

We evaluate TriCo-RAG on HotpotQA (Yang et al., 2018), MuSiQue (Trivedi et al., 2022), and TriviaQA (Joshi et al., 2017). We construct a fixed evaluation set of N=2,976 questions from the **development (dev) splits** (1,000 HotpotQA, 977 MuSiQue, 999 TriviaQA; Appendix B), not from training or test data. Unless noted otherwise, all main experiments use this set with a 2,000-token cap on evidence to the generator.

We report exact match (EM) and F1 as dataset-mean QA metrics (Appendix A.2). To analyze retrieval and evidence utilization, we additionally report **Gold-Supported Generation Rate (GSR)**, the fraction of non-empty predictions whose retained evidence contains a gold answer span; **Answer Hit Rate (AHR)**, the fraction of retained evidence containing any gold answer alias (retrieval-side coverage; Appendix A); and **Evidence Token Count (EvTok)**, the average number of evidence tokens passed to the generator after fusion and TVM selection.

Together, EM and F1 measure answer quality, while GSR, AHR, and EvTok capture evidence coverage and token usage. For supplementary analyses against score-based truncation, we report **EER** and **IER** (Appendix C.1).

For retrieval-side budget analyses (Table 4, Figure C.1), we use a larger pooled run of 6,855 records, of which 6,422 remain after filtering invalid cases. This setting supports retrieval-side budget analysis without repeated generation runs. For the EM/F1 budget trade-off (Section 5.5), we report a separate fixed sample of N=600 instances with complete scores, distinct from the main N=2,976 dev set (Appendix C.2).

4.2 Baselines and Setup

We compare four baselines: (i) **Single-Source** corpus retrieval (CR) (Karpukhin et al., 2020; Lewis et al., 2020), (ii) **Multi-Source** naive fusion (CR&WR) (Agrawal et al., 2025; Tan et al., 2025), (iii) **LLM-Router-RAG** with LLM routing (Guo et al., 2026), and (iv) **Cascade-RAG**, a sequential fallback inspired by coarse-to-fine retrieval and reranking (Nogueira and Cho, 2019; Khattab and Zaharia, 2020). Names in Table 1 match these base-

lines; implementation details are in Appendix E. All methods share the same retrieval, reranking, and generation pipeline for fair comparison.

All methods share hybrid dense–sparse retrieval, cross-encoder reranking, the same evaluation pipeline, and generator (Appendix H), so performance differences reflect retrieval routing and evidence adaptation.

4.3 Environment and Hyperparameters

Experiments are conducted on NVIDIA GPU servers. The retrieval stack uses BGE-small-en-v1.5 for dense retrieval, BM25 for sparse retrieval, and a cross-encoder for candidate rescoring.

We use a fixed 2,000-token evidence cap for the main evaluation; EvTok reports resulting evidence length (Appendix H.2.5). Appendix C.2 shows this budget achieves near-saturated AHR while avoiding diminishing returns at larger budgets.

For routing, we set $\lambda = 0.7$ and the margin threshold $\delta = 0.15$, and we preserve the top-3 evidence items in the warm-start selection. Full hyperparameters are listed in Appendices G and F.

4.4 Generators

We evaluate three generators: Llama 3.1-8B, Qwen3-Max, and GPT-5. All generators use temperature 0.0. Table 1 reports results per generator; other analyses use Qwen3-Max unless noted otherwise (Appendix B.1 summarizes route usage and end-to-end cost for this main run).

4.5 Implementation Details

TriCo-RAG combines LLM-based feature extraction and routing (temperature 0.0). QAR constructs query signals for multi-source retrieval. Routing features, TVM, and web-retrieval details are in Appendices H.2, H.2.5, and H.2.4; the shared evaluation pipeline is in Appendix H. Full prompt templates are omitted here and will be released with the codebase (Appendix I).

5 Results and Analysis

5.1 Main Results

Table 1 reports results on HotpotQA (Yang et al., 2018), TriviaQA (Joshi et al., 2017), and MuSiQue (Trivedi et al., 2022) across three generators. We report EM, F1, Gold-Supported Generation Rate (GSR), and EvTok (evidence tokens passed to the generator after fusion and TVM).

Overall, TriCo-RAG improves accuracy while maintaining efficient evidence utilization through coordinated source routing, query adaptation, and evidence selection. Compared with Single-Source (CR) and Multi-Source (naive CR&WR), TriCo-RAG achieves larger gains on complex datasets with comparable or lower EvTok.

Under Qwen3-Max, TriCo-RAG improves over Multi-Source on HotpotQA, increasing EM/F1 from 29.9/32.4 to 51.5/54.8. On MuSiQue, EM/F1 improves from 7.2/9.4 to 15.4/19.8. These gains suggest that simply enlarging the candidate evidence pool is insufficient; effective multi-source retrieval requires coordinated adaptation rather than naive fusion. On TriviaQA, TriCo-RAG achieves higher EM/F1 (66.1/70.6), the highest GSR (84.9), and lower EvTok than Multi-Source retrieval.

Compared with adaptive baselines, LLM-Router occasionally achieves lower EvTok, but TriCo-RAG more consistently improves answer accuracy and GSR. This suggests that source routing alone is insufficient and that coordinated query adaptation and evidence selection further improve evidence quality and utilization efficiency.

5.2 Ablation Study

Table 2 reports component-level ablations across three datasets. All three components contribute.

Removing SAS reduces answer quality and often increases token usage, highlighting the importance of query-dependent source routing. Removing QAR also causes consistent degradation, suggesting that the original query alone is insufficient for utilizing different retrieval sources. QAR improves evidence recall through retrieval-oriented query variants. Removing TVM causes the largest performance drop, particularly on HotpotQA and TriviaQA, while also increasing evidence tokens. This highlights the importance of token-aware evidence selection under limited budgets.

We further compare TriCo-RAG against fixed-prefix truncation without TVM-style selection. TriCo-RAG achieves higher F1 and EER with lower IER (Appendix C.4, Table C.3), and retention-ratio sweeps show similar trends at matched lengths (Appendix C.5, Fig. C.2).

Overall, the full TriCo-RAG configuration provides the strongest *average* balance between answer quality and token efficiency across datasets, although single-component variants can lead on individual metrics (e.g., Only TVM on MuSiQue GSR). These results suggest that source routing,

Method	HotpotQA				TriviaQA				MuSiQue				Average			
	EM	F1	GSR↑	EvTok↓	EM	F1	GSR↑	EvTok↓	EM	F1	GSR↑	EvTok↓	EM	F1	GSR↑	EvTok↓
Qwen3-Max																
Single-Source	28.2	30.2	43.5	1925	55.0	57.5	68.1	1926	6.6	8.3	25.2	1926	29.9	32.0	45.6	1926
Multi-Source	29.9	32.4	44.6	1963	56.3	59.2	70.2	1978	7.2	9.4	28.9	1856	31.1	33.7	47.9	1932
LLM-Router	50.1	53.4	75.3	**1652**	**66.1**	**70.7**	84.3	**1757**	14.4	19.0	51.2	1525	43.5	47.7	70.3	**1645**
Cascade	49.7	52.7	74.3	1834	63.5	68.3	81.7	1967	14.3	17.9	53.5	1876	42.5	46.3	69.8	1892
TriCo-RAG	**51.5**	**54.8**	**78.8**	1764	**66.1**	70.6	**84.9**	1791	**15.4**	**19.8**	**55.3**	**1416**	**44.3**	**48.4**	**73.0**	1657
GPT-5																
Single-Source	28.2	30.9	46.2	1927	55.0	58.4	70.1	1930	6.5	9.0	24.8	1930	29.9	32.8	47.0	1929
Multi-Source	30.2	34.3	50.3	1927	57.2	59.4	73.4	1930	8.3	10.2	30.4	1928	31.9	34.6	51.4	1928
LLM-Router	47.3	49.2	75.3	1955	61.5	66.5	**84.3**	**1751**	10.2	16.4	51.2	1836	39.7	44.0	70.3	1847
Cascade	45.5	48.2	73.3	1800	59.6	64.7	81.7	1790	11.2	13.0	53.5	1706	38.8	42.0	69.5	1765
TriCo-RAG	**50.7**	**55.3**	**77.4**	**1781**	**66.3**	**71.7**	83.1	1796	**14.2**	**19.5**	**54.2**	**1440**	**43.7**	**48.8**	**71.6**	**1672**
Llama 3.1-8B																
Single-Source	21.3	25.7	79.9	1927	48.1	52.6	92.1	1922	5.0	7.9	51.9	1928	24.8	28.7	74.6	1926
Multi-Source	24.1	28.4	82.0	1928	49.7	53.9	93.2	1923	5.1	8.6	54.6	1929	26.3	30.3	76.6	1927
LLM-Router	34.5	38.1	71.9	1863	56.4	59.4	79.3	1886	5.9	9.9	79.1	1912	32.3	35.8	76.8	1887
Cascade	34.0	37.5	70.7	1852	55.6	58.8	78.1	1875	5.0	8.4	67.9	1902	31.5	34.9	72.2	1876
TriCo-RAG	**35.2**	**39.7**	**92.0**	**1702**	**56.6**	**60.7**	**94.1**	**1730**	**6.9**	**10.9**	**82.5**	**1754**	**32.9**	**37.1**	**89.5**	**1729**

Table 1: Main results across retrieval settings on HotpotQA, TriviaQA, and MuSiQue. Last four columns report macro-averaged results. EvTok denotes evidence tokens. Best and second-best are bold and underlined, respectively.

Component	Hotpot				MuSiQue				TriviaQA				Average			
	EM	F1	GSR↑	EvTok↓	EM	F1	GSR↑	EvTok↓	EM	F1	GSR↑	EvTok↓	EM	F1	GSR↑	EvTok↓
TriCo-RAG (Full)	**57.8**	**61.5**	**84.8**	**1745**	**17.6**	**22.3**	55.8	**1345**	69.7	74.0	**85.2**	**1740**	**48.4**	**52.6**	**75.3**	**1610**
w/o SAS	55.5	59.3	82.5	1800	15.5	20.2	53.5	1786	67.5	71.9	84.0	1800	46.2	50.5	73.3	1795
w/o QAR	56.0	59.3	82.5	1757	17.5	21.9	56.5	1365	69.0	72.9	83.5	1786	47.5	51.4	74.2	1636
w/o TVM	50.5	53.3	75.0	1882	14.5	19.4	54.5	1453	64.5	68.3	80.5	1911	43.2	47.0	70.0	1749
Only SAS	54.5	57.6	78.0	1876	17.3	21.1	54.6	1438	**70.5**	**74.5**	85.0	1914	47.4	51.1	72.5	1743
Only QAR	53.0	55.3	77.5	1927	12.5	16.6	44.5	1932	65.5	69.3	81.0	1930	43.7	47.1	67.7	1930
Only TVM	56.5	60.0	82.5	1791	15.5	20.8	**58.0**	1759	68.5	70.3	84.5	1801	46.8	50.4	75.0	1784

Table 2: Component-level ablation of TriCo-RAG under component removal (w/o) and single-component settings (N=600 dev subset, disjoint from the main N=2,976 set; Appendix B). ↑ higher is better for GSR; ↓ lower is better for EvTok.

Source	Method	EM	F1	GSR↑	AHR↑
Single	Baseline (w/o QAR)	30.1	32.9	44.2	40.4
	TriCo-RAG	**44.0**	**49.1**	**64.1**	**61.4**
Multi	Baseline (w/o QAR)	33.0	34.0	51.4	40.4
	TriCo-RAG	**48.5**	**53.9**	**67.5**	**64.1**

Table 3: QAR results with Qwen3-Max under matched retrieval and evidence budgets. Macro-averaged over HotpotQA, TriviaQA, and MuSiQue.

query adaptation, and evidence selection are complementary rather than isolated mechanisms. Appendix D (Table D.1) provides instance-level comparisons under a 3,000-token evidence budget.

5.3 QAR Analysis

To analyze the effect of QAR, we compare TriCo-RAG against a matched baseline without QAR under the same retrieval setting and evidence budget. Single-source retrieval uses a corpus retriever, while multi-source retrieval combines corpus and web retrieval.

The w/o-QAR baseline disables query variants only: SAS routing and TVM selection remain active under the same reranking pipeline. Matched evidence budgets ensure that differences in EM, F1, GSR, and AHR reflect query-side retrieval cues rather than retrieving or retaining additional tokens.

Table 3 reports macro-averaged results on HotpotQA, TriviaQA, and MuSiQue with Qwen3-Max. QAR consistently improves performance in both settings. Under single-source retrieval, TriCo-RAG improves AHR from 40.4 to 61.4, GSR from 44.2 to 64.1, and EM/F1 from 30.1/32.9 to 44.0/49.1; under multi-source retrieval, AHR rises from 40.4 to 64.1, GSR from 51.4 to 67.5, and EM/F1 from 33.0/34.0 to 48.5/53.9. These results suggest that query-adaptive reformulation improves how retrieval sources are utilized under fixed budgets, benefiting downstream evidence selection.

5.4 TVM Budget Sensitivity

We analyze evidence utilization under different token budgets using AHR as a retrieval-side coverage

Budget	Hit Count	AHR (%)	Marginal Gain (%)
500	3,149	49.0	-
1,000	3,526	54.9	+5.9
1,400	3,731	58.1	+3.2
1,600	3,822	59.5	+1.4
2,000	**4,082**	**63.6**	**+4.1**
2,500	4,272	66.5	+2.9
2,700+	4,275	66.6	0.0 (saturated)

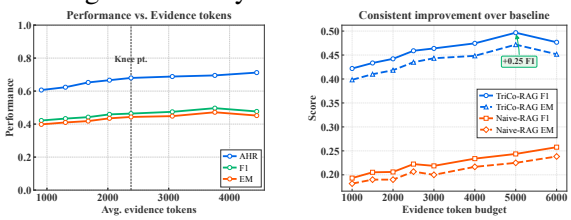
Table 4: AHR across token budgets (6,422 samples).

metric (Appendix C.2).

5.4.1 AHR vs. Evidence Budget

Table 4 reports AHR under representative evidence budgets. AHR increases rapidly at low and moderate budgets before gradually saturating. At 2,000 tokens, AHR reaches 63.6%, or 95.5% of the maximum value at 2,700+ tokens, indicating diminishing returns from additional evidence.

Figure C.1 in Appendix C.2 plots the continuous AHR–budget curve for the same 6,422-sample setting. These results support moderate evidence budgets that preserve most retrieval coverage while avoiding unnecessary token costs.



(a) AHR/F1/EM vs. evidence budget. (b) TriCo vs. Multi-Source.

Figure 3: Budget sweep (N=600). **(a)** AHR/F1/EM vs. evidence budget; **(b)** TriCo-RAG vs. Multi-Source.

5.5 Evidence Budget and Answer Quality

We sweep evidence budgets on N=600 instances (full results in Appendix C.2, Table C.1). Figure 3 shows how evidence length affects answer quality. Panel (a) shows AHR, F1, and EM rising at low and moderate lengths before saturating, consistent with Table 4 and diminishing returns once useful evidence is covered. Panel (b) compares F1 and EM under matched budgets: TriCo-RAG outperforms Multi-Source by roughly 0.25 F1 across budgets, indicating gains from coordinated evidence selection rather than larger budgets.

5.5.1 Marginal Gain Analysis

Appendix C.6 reports a heatmap of average marginal gains per 100 realized evidence tokens across low-, mid-, and high-budget ranges. Marginal gains concentrate at low budgets and diminish as the budget increases. F1 and EM saturate earlier than AHR, suggesting that retrieval-side coverage may continue improving after answer quality plateaus. These results further motivate TVM: as later tokens contribute smaller gains, evidence selection should adapt to marginal utility rather than retaining more evidence.

Top-K	Hit Count	AHR (%)	Cumulative AHR (%)
1	1,892	29.6	46.1
2	2,476	40.9	60.3
3	2,804	48.5	68.3
4	2,965	51.3	72.2
5	3,099	53.3	75.4
6	3,225	59.4	78.5
7	3,430	69.4	83.5
8	3,529	73.5	85.9
9	3,626	79.6	88.3
10	3,693	81.2	89.9

Table 5: AHR by Top-K. Cumulative percentages are computed over answer-supported samples.

5.5.2 Supporting Evidence Distribution

We analyze where answer-supporting evidence appears in the ranked retrieval results to understand the design of TVM under constrained budgets.

Table 5 shows a strongly front-loaded distribution. The top-3 retrieved items achieve 48.5% AHR and cover 68.3% of all answer hits, while the top-10 reaches 81.2% AHR with 89.9% cumulative coverage. These results indicate that most answer-supporting evidence appears at shallow ranks. Appendix C.7 further visualizes the Top-K distribution, showing that AHR rises rapidly at small K and that gold-first hits are concentrated near the top ranks. These findings support the design of TVM, which combines a high-precision warm start with token-aware evidence selection, allowing evidence filtering to adapt to retrieval cues introduced during earlier query reformulation.

6 Conclusion

We present TriCo-RAG, an adaptive framework for multi-source RAG that coordinates source selection, query adaptation, and evidence selection. Experiments across datasets and generators show consistent gains in answer accuracy with more efficient evidence use.

Analyses further show that QAR improves evidence recall, TVM is effective under constrained budgets, and SAS better matches retrieval sources to query needs. Overall, effective multi-source RAG requires adaptive coordination rather than simply enlarging the evidence pool. Future work includes learned routing, broader generation tasks, and multilingual or domain-specific adaptation.

Limitations

TriCo-RAG relies on heuristic coordination for routing and evidence selection. In particular, TVM optimizes query-feature coverage, which may not always align with downstream answer quality.

Our evaluation is limited to English factual QA and depends on the quality and coverage of the underlying retrieval sources. Broader domains, languages, and learned coordination mechanisms are left to future work.

References

Rishabh Agrawal, Murtaza Asrani, Hadi Youssef, and Apurva Narayan. 2025. SCMRAG: Self-corrective multihop retrieval-augmented generation system for LLM agents. In *Proceedings of the 24th International Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2025)*, pages 50–58, Detroit, Michigan, USA. IFAAMAS.

Paul J. L. Ammann, Jonas Golde, and Alan Akbik. 2025. Question decomposition for retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 497–507, Vienna, Austria. Association for Computational Linguistics.

Chia-Yuan Chang, Zhimeng Jiang, Vineeth Rakesh, Menghai Pan, Chin-Chia Michael Yeh, Guanchu Wang, Mingzhi Hu, Zhichao Xu, Yan Zheng, Mahashweta Das, and Na Zou. 2025. Main-rag: Multi-agent filtering retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2607–2622, Vienna, Austria. Association for Computational Linguistics.

Yiqun Chen, Lingyong Yan, Weiwei Sun, Xinyu Ma, Yi Zhang, Shuaiqiang Wang, Dawei Yin, Yiming Yang, and Jiaxin Mao. 2025. Improving retrieval-augmented generation through multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, and Furu Wei. 2024. xRAG: Extreme context compression for retrieval-augmented generation with one token. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

Guanting Dong, Jiajie Jin, Xiaoxi Li, Yutao Zhu, Zhicheng Dou, and Ji-Rong Wen. 2025. Rag-critic: Leveraging automated critic-guided agentic workflow for retrieval augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3551–3578, Vienna, Austria. Association for Computational Linguistics.

Minghao Guo, Qingcheng Zeng, Xujiang Zhao, Yanchi Liu, Wenchao Yu, Mengnan Du, Haifeng Chen, and Wei Cheng. 2026. DeepSieve: Information sieving via LLM-as-a-knowledge-router. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 3054–3077, Rabat, Morocco. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, pages 3929–3938, Virtual. PMLR.

Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A. Rossi, Subhabrata Mukherjee, Xianfeng Tang, Qi He, Zhigang Hua, Bo Long, Tong Zhao, Neil Shah, Amin Javari, Yinglong Xia, and Jiliang Tang. 2025. Retrieval-augmented generation with graphs (graphrag). *arXiv preprint arXiv:2501.00309*.

Jeongyeon Hwang, Junyoung Park, Hyejin Park, Dongwoo Kim, Sangdon Park, and Jungseul Ok. 2025. Retrieval-augmented generation with estimation of source reliability. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 34279–34303, Suzhou, China. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 609–623, Online. Association for Computational Linguistics.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Hailey Joren, Jianyi Zhang, Chun-Sung Ferng, Da-Cheng Juan, Ankur Taly, and Cyrus Rashtchian. 2025. Sufficient context: A new lens on retrieval-augmented generation systems. In *Proceedings of the International Conference on Learning Representations (ICLR 2025)*.

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.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. 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 2020)*, pages 6769–6781, Online. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 39–48, New York, NY, USA. ACM.

Patrick S. H. 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 2020)*, pages 9459–9474, Red Hook, NY, USA. Curran Associates, Inc.

Mingda Li, Xinyu Li, Yifan Chen, Wenfeng Xuan, and Weinan Zhang. 2024. Unraveling and mitigating retriever inconsistencies in retrieval-augmented large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4833–4850, Bangkok, Thailand. Association for Computational Linguistics.

Rui Li, Liyang He, Qi Liu, Zheng Zhang, Heng Yu, Yuyang Ye, Linbo Zhu, and Yu Su. 2025. Unirag: Unified query understanding method for retrieval augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14163–14178, Vienna, Austria. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14(1):265–294.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 1–10, Hong Kong, China. Association for Computational Linguistics.

OpenAI. 2024. GPT-4 technical report. *Preprint*, arXiv:2303.08774.

Pranoy Panda, Raghav Magazine, Chaitanya Devaguptapu, Sho Takemori, and Vishal Sharma. 2025. Adaptive LLM routing under budget constraints. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 23934–23949, Suzhou, China. Association for Computational Linguistics.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. Colbertv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, Seattle, WA, USA. Association for Computational Linguistics.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*. OpenReview.net.

Yash Saxena, Ankur Padia, Mandar S. Chaudhary, Kalpa Gunaratna, Srinivasan Parthasarathy, and Manas Gaur. 2025. Ranking free RAG: Replacing reranking with selection in RAG for sensitive domains. *arXiv preprint arXiv:2505.16014*.

Wenyuan Shi, Sewon Min, Wen-tau Yih, and et al. 2023. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Jiashuo Sun, Xianrui Zhong, Sizhe Zhou, and Jiawei Han. 2025. DynamicRAG: Leveraging outputs of large language model as feedback for dynamic reranking in retrieval-augmented generation. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*.

Maxim Sviridenko. 2004. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41–43.

Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, Liming Zhu, and Wenjie Zhang. 2025. HydraRAG: Structured cross-source enhanced large language model reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14431–14459, Suzhou, China. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu,

Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

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.

Yile Wang, Peng Li, Maosong Sun, and Yang Liu. 2023. Self-knowledge guided retrieval augmentation for large language models.

Zihan Wang, Zihan Liang, Zhou Shao, Yufei Ma, Huangyu Dai, Ben Chen, Lingtao Mao, Chenyi Lei, Yuqing Ding, and Han Li. 2025. Infogain-rag: Boosting retrieval-augmented generation through document information gain-based reranking and filtering. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7190–7204, Suzhou, China. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed 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*, pages 24824–24837, Red Hook, NY, USA. Curran Associates, Inc.

Jiaqi Wei, Hao Zhou, Xiang Zhang, Di Zhang, Zijie Qiu, Wei Wei, Jinzhe Li, Wanli Ouyang, and Siqi Sun. 2025. Retrieval is not enough: Enhancing RAG reasoning through test-time critique and optimization. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*.

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 *Proceedings of the Eleventh International Conference on Learning Representations (ICLR 2023)*, Kigali, Rwanda. OpenReview.net.

Xiaozhou You, Yahui Luo, and Lihong Gu. 2025. Ms-rag: Simple and effective multi-semantic retrieval-augmented generation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 22609–22625, Suzhou, China. Association for Computational Linguistics.

Junchi Yu, Yujie Liu, Jindong Gu, Philip H. S. Torr, and Dongzhan Zhou. 2025. Can knowledge-graph-based retrieval augmented generation really retrieve what you need? In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*.

Jiarui Zhang, Xiangyu Liu, Yong Hu, Chaoyue Niu, Fan Wu, and Guihai Chen. 2025. RAGRouter: Learning to route queries to multiple retrieval-augmented language models. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*.

Yujia Zhou, Zheng Liu, Jiajie Jin, Jian-Yun Nie, and Zhicheng Dou. 2024. Metacognitive retrieval-augmented large language models. In *Proceedings of the ACM Web Conference 2024*, pages 1453–1463, New York, NY, USA. Association for Computing Machinery.

Appendix

A Evaluation Metrics

A.1 Answer Hit Rate (AHR)

Answer Hit Rate (AHR) measures whether the evidence context passed to the generator contains at least one normalized gold answer string. After applying the same normalization used for EM, we mark a *hit* if any gold alias appears as a contiguous substring in the retained evidence context; otherwise the instance is marked as a *miss*. AHR is computed on instances with non-empty gold aliases and retained evidence. It complements EM/F1 by measuring whether answer-bearing evidence is successfully delivered to the generator.

A.2 QA Metrics

We also report standard QA and evidence-efficiency metrics:

- **EM**: exact match after normalization.
- **F1**: token-level F1 (max over multiple gold answers when applicable).
- **GSR**: the fraction of non-empty generations for which the retained evidence contains a gold answer span.
- **EvTok**: evidence tokens passed to the generator after fusion and TVM (Tables 1 and 2).

Aggregation and reporting. All reported results are obtained from one full evaluation pass per method on the fixed evaluation IDs (dataset sizes are given in Appendix B). **EM** denotes the mean exact-match rate (%), and **F1** denotes the mean token-level F1 (%), using the maximum score over gold aliases for each question. **GSR** and **AHR** are averaged instance-level hit rates, while **EvTok** denotes the mean evidence-token count per question. The **Average** columns in Tables 1 and 2 macro-average dataset-level results across HotpotQA, MuSiQue, and TriviaQA with equal dataset weights. Results are reported from single deterministic runs with temperature 0.0 and fixed random seeds (Appendix I).

B Dataset Sampling

We build fixed dataset pools in unified JSONL format from each benchmark's official development (dev) split (not training or held-out test). Each record stores the original benchmark identifier, question text, normalized gold answers (primary span and aliases), and gold-evidence metadata.

Main evaluation set (N=2,976). The main set uses the *complete dev pool* for each benchmark rather than re-sampling from training data: HotpotQA dev (n=1,000), TriviaQA dev (n=999 unique IDs), and MuSiQue dev (n=977), totaling 2,976 instances. Every evaluation ID appears in the corresponding dev pool with a one-to-one match, so results can be reproduced from the released ID lists without random subsampling noise. This protocol prioritizes a stable, fully enumerated cross-dataset comparison under a shared pipeline; it is not directly comparable to official leaderboard numbers on unseen test splits unless those IDs are evaluated separately.

For other analyses that require subsampling (e.g., pilot runs), pools support proportional allocation with target size T:

$$n_i = \max\left(\text{min_per_dataset}, \left\lfloor \frac{N_i^{\text{pool}}}{\sum_j N_j^{\text{pool}}} \times T \right\rfloor\right),$$

with a default minimum of 50 instances per dataset.

Ablation subset. Component ablations (Table 2) use a separate balanced development (dev) sample of N=600 questions (200 per dataset), drawn from the same dev pools but with IDs disjoint from the main N=2,976 evaluation set. This subset is not a subset of the main dev run; it supports controlled component comparisons without repeating full-scale generation on all 2,976 instances.

B.1 Run-Level Statistics (Main Evaluation)

Tables B.1 and B.2 summarize feature and route distributions and end-to-end cost statistics on the Qwen3-Max main evaluation (N=2,976).

Route outcome analysis. On the main run, EM by executed route is 49.3% for CR (n=2,552) and 15.8% for CR&WR (n=424). CR&WR concentrates on higher-hop and higher-evidence-count questions (Table C.2a), which are intrinsically harder; the routing pattern reflects both the SAS prior and query difficulty rather than uniform random misrouting.

C Supplementary Diagnostics

This section reports auxiliary routing, evidence-quality, and budget analyses (vs. a score-based truncation baseline).

Feature	Count	Share (%)
Reasoning steps (h)		
1-hop	627	21.1
2-hop	1,906	64.0
3-hop	390	13.1
4-hop+	53	1.8
Evidence count (c)		
single	584	19.6
two	1,893	63.6
few	439	14.8
several / many	60	2.0
Executed route		
CR	2,552	85.8
CR&WR	424	14.2

Table B.1: Feature and route distributions for TriCo-RAG (Qwen3-Max main run). WR-only routes did not occur.

Per-question statistic	Value
Mean LLM API calls	2.25
Mean web searches	0.14
Mean generator evidence tokens	2,822
Mean end-to-end latency (ms)	4,353
Stage latency (ms, mean)	
Feature extraction	1,642
Route planning	4
Retrieval	2,172
Fusion (rerank + compress)	7,122
Answer generation	409
Per-dataset EM (%)	
HotpotQA	51.5
TriviaQA	66.1
MuSiQue	15.4

Table B.2: Cost and quality summary for TriCo-RAG (Qwen3-Max main run). Evidence-token counts use the same accounting as Table 1.

C.1 Diagnostic Metrics

ERR, EER, and IER. The evidence retention ratio $\mathrm{ERR} = L_{\mathrm{ret}}/L_{\mathrm{orig}}$ measures the proportion of retrieved candidates retained before budget enforcement. We sweep $\mathrm{ERR} \in \{0.25, 0.50, 0.75, 1.00\}$ with L_{orig}=1,000 in Appendix C.5 (distinct from the TVM compress-budget share in Table F.1). **Effective Evidence Ratio (EER)** is $\mathrm{EER} = L_{\mathrm{effective}}/L_{\mathrm{ret}}$ and **Ineffective Evidence Ratio (IER)** is $\mathrm{IER} = 1 - \mathrm{EER}$, where $L_{\mathrm{effective}}$ is retained evidence identified as answer-supporting under the same evidence-matching protocol used for GSR/AHR diagnostics.

C.2 AHR vs. evidence budget

Table C.1 and Figure C.1 extend the budget analysis (Section 5.5; N=600 for EM/F1, 6,422 for retrieval-side AHR).

Budget	Avg. Tok	AHR	EM	F1	G_{AHR}	G_{EM}	G_{F1}
TriCo-RAG							
1,000	908	60.7	39.8	42.2	–	–	–
1,500	1,304	62.4	41.0	43.3	0.43	0.30	0.29
2,000	1,677	65.3	41.8	44.2	0.78	0.22	0.23
2,500	2,027	66.6	43.5	45.9	0.39	0.48	0.48
3,000	2,381	68.0	44.3	46.4	0.39	0.23	0.14
4,000	3,059	68.8	44.8	47.4	0.13	0.07	0.16
5,000	3,758	69.5	47.2	49.7	0.10	0.33	0.32
6,000	4,445	71.2	45.2	47.7	0.25	-0.29	-0.29
Truncation Baseline							
1,000	930	33.9	18.2	19.3	–	–	–
1,500	1,429	38.0	19.0	20.5	0.82	0.17	0.24
2,000	1,931	39.5	19.0	20.6	0.31	0.00	0.01
2,500	2,431	42.4	20.7	22.2	0.58	0.33	0.33
3,000	2,927	44.5	20.0	21.9	0.41	-0.14	-0.07
4,000	3,925	47.1	21.7	23.4	0.26	0.17	0.15
5,000	4,915	49.7	22.5	24.4	0.27	0.08	0.10
6,000	5,899	53.1	23.8	25.8	0.34	0.14	0.14

Table C.1: Budget sweep (N=600). Gain columns: marginal Δ per 100 realized tokens between adjacent budgets.

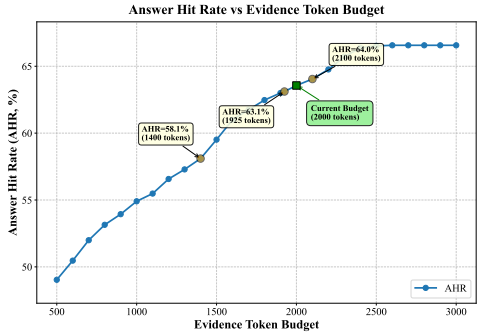


Figure C.1: AHR vs. evidence budget (6,422 samples; Table 4). At 2,000 tokens, AHR is 95.5% of the post-2,700 maximum.

C.3 SAS Route Distribution

Tables C.2a and C.2b summarize SAS route usage (TriviaQA favors CR; MuSiQue uses more CR&WR). Most 2-hop instances are routed to CR because the feature prior assigns high mass to CR for shallow multi-hop questions with corpus-sufficient evidence (Table H.1), while 3-hop and 4-hop+ questions more often trigger CR&WR when bridging or web supplementation is needed. Reported routes follow the three-way label set $\{\mathrm{CR}, \mathrm{WR}, \mathrm{CR\&WR}\}$ used in the main experiments.

C.4 SAS Variants vs. Truncation

Table C.3 compares TriCo-RAG SAS variants (CR/WR/CR&WR) with a truncation baseline (fixed prefix, no TVM).

C.5 Robustness under ERR

We sweep $\mathrm{ERR} = L_{\mathrm{ret}}/L_{\mathrm{orig}}$ and compare TriCo-RAG with score-based truncation (no TVM).

Hops	Tot.	CR	CR&WR
1-hop	627	100.0%	0.0%
2-hop	1,906	99.8%	0.2%
3-hop	390	5.9%	94.1%
4-hop+	53	0.0%	100.0%
Tot.	**2,976**	**85.7%**	**14.3%**

(a) By hops

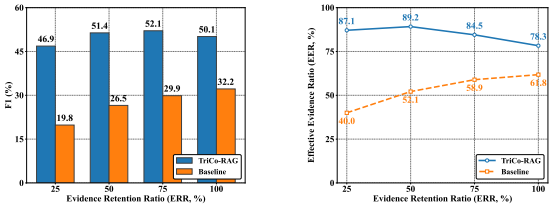
Dataset	Tot.	CR	CR&WR
HotpotQA	1,279	96.5%	3.5%
MuSiQue	638	61.4%	38.6%
TriviaQA	937	98.8%	1.2%
Tot.	**2,854**	**89.4%**	**10.6%**

(b) By dataset

Table C.2: SAS route distributions. **Tot.**: question count; route columns: row %. Dataset total (2,854) < hop total (2,976) when hop labels are missing.

Method	F1↑	EER↑	IER↓
TriCo-RAG (CR)	0.62	0.92	0.08
TriCo-RAG (WR)	0.49	0.86	0.14
TriCo-RAG (CR&WR)	0.53	0.90	0.10
Truncation Baseline	0.32	0.62	0.38

Table C.3: SAS routes vs. truncation (F1/EER↑, IER↓).



(a) F1 vs. ERR. (b) EER vs. ERR.

Figure C.2: F1 and EER vs. ERR (25–100%). TriCo-RAG leads at every setting; gains are largest under strict retention (Appendix C.1).

C.6 Marginal-Gain Heatmap

Following the budget sweep in Appendix C.2, Figure C.3 summarizes *average* marginal gains per 100 realized evidence tokens for AHR, F1, and EM across low-, mid-, and high-budget regimes (Section 5.5). Gains are largest at low budgets and taper as budgets grow; F1 and EM plateau earlier than AHR, indicating that later tokens mainly extend retrieval-side coverage rather than answer quality.

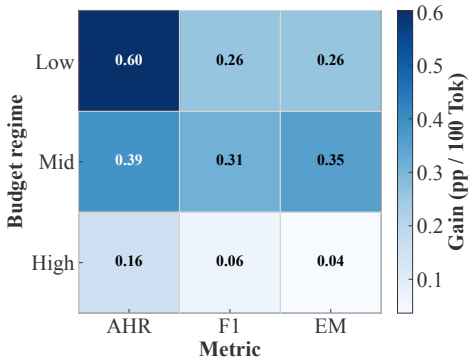
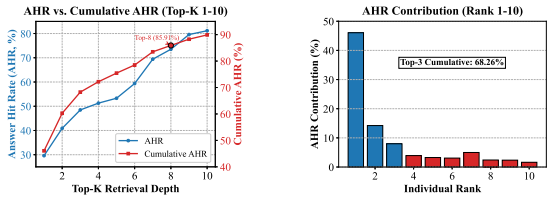


Figure C.3: Marginal gains (pp per 100 tokens) for low (1k–2k), mid (2.5k–4k), and high (5k–6k) budgets.

C.7 Top-K Rank Visualization

Figure C.4 complements Table 5 in the main text. Panel (a) shows how prefix-truncation AHR grows with Top-K; panel (b) shows where gold-first hits concentrate among answer-supported questions. Together, the curves confirm a strongly front-loaded rank distribution, consistent with the shallow-rank statistics in Table 5.



(a) AHR vs. Top-K. (b) Gold-first rank mass.

Figure C.4: Top-K AHR (a) and gold-first rank mass among hits (b).

Most answer-bearing evidence therefore appears within the top few ranks, supporting TVM's warm-start prefix and token-aware selection under fixed budgets.

D Case Studies

D.1 MuSiQue Case Study

Question: When did the singer of Love Lifts Us Up Where We Belong record With a Little Help From My Friends?
SAS: CR&WR; web retrieval augments corpus evidence.
QAR: Q_o keeps the original question; Q_b links the ballad to its vocalists and the Beatles cover; Q_e centers Joe Cocker and the album title; Q_a targets the recording-time answer type.
Outcome: TriCo-RAG reaches EM=1 with EvTok 537 vs. EM=0 and EvTok 2,915 for Multi-Source (Δ=2,378 tokens).

Table D.1 summarizes five instances (HotpotQA, MuSiQue, TriviaQA) at a 3,000-token budget under the same protocol.

Instance ID	Dataset	EM		EvTok↓		ΔEvTok
		Multi	TriCo	Multi	TriCo	
2hop__64563_86874	MuSiQue	0	1	2,915	537	2,378
5a8c52565542995e66a475ba	HotpotQA	0	1	2,944	939	2,005
triviaqa_376853	TriviaQA	0	1	2,995	2,423	572
5ae806225542994a481bbe8b	HotpotQA	0	1	2,997	2,467	530
triviaqa_499058	TriviaQA	0	1	2,965	2,443	522

Table D.1: Case studies at 3,000-token budget.
Multi-Source: naive CR&WR fusion baseline; **TriCo**: TriCo-RAG.
$\Delta\text{EvTok} = \text{EvTok}_{\text{Multi-Source}} - \text{EvTok}_{\text{TriCo}}$.

E Baseline Details

To ensure fair comparison, all baseline methods share the same retrieval backbones, corpus, evidence budget, and evaluation protocol. This section

provides detailed implementation specifications for each baseline method evaluated in Table 1.

E.1 LLM-Router-RAG

LLM-Router-RAG follows the LLM-as-router paradigm of DeepSieve (Guo et al., 2026). Given a query, an LLM selects one route from {CR, WR, CR&WR} using a fixed routing prompt, with deterministic decoding (temperature $= 0.0$). The selected route is executed directly; for CR&WR, corpus and web retrieval run in parallel and are merged with the same fusion pipeline used by the Multi-Source baseline.

E.2 Cascade-RAG

Cascade-RAG is a heuristic sequential fallback baseline inspired by coarse-to-fine retrieval and reranking principles. It first executes CR, then triggers WR when CR confidence is insufficient (low passage count, low average score, or no high-similarity passage). Returned WR evidence is merged with CR evidence using the same fusion mechanism as the Multi-Source baseline. Thresholds are fixed for all datasets and are not tuned per method.

E.3 Single-Source Corpus Retrieval (CR)

The single-source baseline retrieves only from the static corpus (CR), following standard open-domain RAG with dense–sparse hybrid retrieval (Karpukhin et al., 2020; Lewis et al., 2020). It uses the same retrieval stack, reranker, and evidence budget as other methods, without web retrieval or adaptive source selection.

E.4 Multi-Source (Naive CR&WR Fusion)

The Multi-Source baseline (Table 1) performs naive multi-source fusion (CR&WR) without adaptive source selection, in line with pooled multi-source RAG settings (Agrawal et al., 2025; Tan et al., 2025). Corpus and web candidates are merged into a shared rerank pool using the same candidate normalization and rank-based scoring interface described in Appendix H. Duplicate candidates are merged by canonical chunk ID while retaining the maximum fused score before reranking and evidence selection.

E.5 Shared Hyperparameters

All baseline methods share the following hyperparameters to ensure fair comparison:

- **Top-k retrieval**: candidates are retrieved through the shared retrieval interface and retained according to the common rerank-pool configuration before reranking;
- **Chunk size**: 384 tokens per passage/document chunk (overlap 48);
- **Max tokens (evidence budget)**: 2,000 tokens cap on evidence passed to the generator; TVM greedy selection uses a compress budget of 1,600 tokens (80% of the cap; Table F.1);
- **Reranking**: Cross-encoder reranker (ms-marco-MiniLM-L6-v2) applied to the shared rerank pool before evidence selection;
- **LLM temperature**: 0.0 for deterministic generation and routing;
- **Dense retrieval**: BGE-small-en-v1.5 embeddings (384-dimensional) with IVF index ($nlist = 4096$, $nprobe = 32$);
- **Sparse retrieval**: BM25 with standard English tokenization, stopword removal, and stemming;
- **Retrieval scoring**: dense–sparse retrieval uses shared weights $w_d = 0.6$, $w_b = 0.4$, followed by the rank-based fused scoring interface in Appendix H.

These hyperparameters are fixed across all baseline methods and are not tuned individually for each method. The same hyperparameter configuration is used for TriCo-RAG, as reported in Table F.1.

E.6 Tuning & Fairness

All methods share the same development (dev) evaluation set, fixed hyperparameters after selection, the same retrieval backbones, corpus, generator, evaluation pipeline, and evidence budget. This protocol is intended to make differences in Table 1 primarily attributable to source selection (SAS), query-signal construction, and evidence selection rather than implementation-specific tuning.

F Hyperparameters

We report the key hyperparameters used in TriCo-RAG across all experiments. These settings were selected through preliminary tuning on a development (dev) split and remain fixed throughout our evaluation. Table F.1 summarizes the configuration.

G Experimental Environment

This section summarizes the environment used for index construction and inference:

Parameter	Setting
Evidence cap (generator input)	2,000 tokens
TVM compress-budget share	0.80
Min evidence guarantee	3
Gain threshold	0.0003
SAS mixture weight λ	0.7
SAS margin δ	0.15
Hybrid weights (w_d/w_b)	0.6 / 0.4
Query signal weights (Q_o, Q_b, Q_e, Q_a)	0.5 / 0.2 / 0.2 / 0.1
LLM temperature	0.0

Table F.1: TriCo-RAG hyperparameter settings. TVM compress-budget share uses 80% of the evidence cap (≈1,600 tokens for greedy TVM).

- **Index construction**: GPU with 48GB+ VRAM (e.g., A100/H100) to build the dense IVF index; BM25 indexes built with a Lucene-based pipeline.
- **Inference**: Preloaded dense and sparse indexes to avoid first-query I/O overhead; retrieval performed through a unified dense–sparse interface.
- **Retrievers**: dense retrieval with FAISS IVF; sparse retrieval with BM25 (Pyserini/Lucene).

H Implementation Details

H.1 QAR Architecture

Following Section 3.3, QAR constructs up to four query signals that are fused into a unified retrieval representation:

- **Original Query** (Q_o): Uses the original question text as-is.
- **Bridge Query** (Q_b): optional bridge variant when $\mathrm{BridgeTrigger}(f)$; see $\mathrm{GenerateBridgeQuery}$ in Appendix H.1.2.
- **Entity-Augmented Query** (Q_e): Query enriched with extracted named entities to improve entity matching.
- **Answer-Oriented Query** (Q_a): Query reformulated to focus on answer type and expected format.

H.1.1 Pseudocode for SAS+QAR

H.1.2 Helpers for Alg. 2

To keep the main pseudocode concise, we define here the helper functions used by SAS+QAR.

Algorithm 2: Source-Adaptive Selection and Query-Adaptive Reformulation

Input: query q; source set $\mathcal{R} = \{\mathrm{CR}, \mathrm{WR}, \mathrm{CR\&WR}\}$; features f; mixture weight $\lambda \in [0, 1]$; margin $\delta > 0$; fusion size M
Output: selected source(s) $r^\star$; query signals $\{Q_o, Q_b, Q_e, Q_a\}$

for $r \in \mathcal{R}$ **do**
  $s(r) \leftarrow \lambda\, s_{\text{prior}}(r \mid f) + (1-\lambda)\, s_{\text{llm}}(r \mid q)$;
Let $r_{(1)}, r_{(2)}$ be the top-1 and top-2 sources by $s(r)$;
if $s(r_{(1)}) - s(r_{(2)}) \geq \delta$ **then**
  $r^\star \leftarrow \{r_{(1)}\}$;
else
  $r^\star \leftarrow$ the top-M sources in $\mathcal{R}$ by $s(r)$;
$Q_o \leftarrow q$, ;
$Q_e \leftarrow \mathrm{EntityAugment}(q)$, ;
$Q_a \leftarrow \mathrm{AnswerTypeReformulate}(q)$;
if $\mathrm{BridgeTrigger}(f)$ **then**
  $Q_b \leftarrow \mathrm{GenerateBridgeQuery}(q)$;
else
  $Q_b \leftarrow \emptyset$;
return $r^\star, \{Q_o, Q_b, Q_e, Q_a\}$;

BridgeTrigger(f)**. Input:** feature tuple $f = (h, d, c, t)$. **Output:** binary decision in {`true`, `false`}. **Rule:** return `true` when the query is classified as multi-hop ($h \in \{\text{2-hop}, \text{3-hop}, \text{4-hop+}\}$) and retrieval difficulty is medium/hard ($d \in \{\text{medium}, \text{hard}\}$); otherwise return `false`. **Purpose:** activate bridge-style reformulation only when intermediate evidence is likely needed.

GenerateBridgeQuery(q)**. Input:** original query q. **Output:** bridge query Q_b (or $\emptyset$ if invalid). **Procedure:** invoke an LLM with a bridge-query prompt (temperature $= 0.0$), then post-process with whitespace/quote cleanup and length validation (3–200 tokens). **Fallback:** if generation fails validation, set $Q_b \leftarrow \emptyset$ and continue with $\{Q_o, Q_e, Q_a\}$.

EntityAugment(q)**. Input:** original query q. **Output:** entity-augmented query Q_e. **Rule:** extract salient entities from q and append normalized entity cues to improve entity-centric retrieval recall.

AnswerTypeReformulate(q)**. Input:** original query q. **Output:** answer-oriented query Q_a. **Rule:** reformulate q toward expected answer type/format (e.g., person, location, date, numeric) to improve target-oriented retrieval.

H.2 SAS Features and Manual Prior Table

For reproducibility, we report the SAS feature schema and the hand-crafted prior policy $\mathrm{TablePrior}(r; h, d, c)$ used in Section 3.2. For brevity, Table H.1 shows only the $h = 2$-hop and $h = 4$-hop+ slices.

Feature design rationale. h is derived from the extractor's reasoning-step count and buckets multi-

h (steps)	d (difficulty)	c (evidence)	CR	WR	CR&WR
2-hop	easy	single (1)	0.70	0.20	0.10
		multi (2-7)	0.60	0.20	0.20
		many (8+)	0.50	0.20	0.30
	medium	single (1)	0.55	0.25	0.20
		multi (2-7)	0.45	0.25	0.30
		many (8+)	0.35	0.25	0.40
	hard	single (1)	0.40	0.35	0.25
		multi (2-7)	0.30	0.35	0.35
		many (8+)	0.25	0.35	0.40
4-hop+	easy	single (1)	0.30	0.10	0.60
		multi (2-7)	0.25	0.10	0.65
		many (8+)	0.20	0.10	0.70
	medium	single (1)	0.25	0.15	0.60
		multi (2-7)	0.20	0.15	0.65
		many (8+)	0.15	0.15	0.70
	hard	single (1)	0.20	0.20	0.60
		multi (2-7)	0.15	0.20	0.65
		many (8+)	0.10	0.25	0.65

Table H.1: Hand-crafted SAS prior policy (excerpt). We show only the $h = 2$-hop and $h = 4$-hop+ slices for brevity.

hop depth (1-hop through 4-hop+). d reflects whether a static Wikipedia corpus is likely sufficient (easy) vs. needing noisier or fresher web evidence (hard). c buckets how many distinct supporting passages the question requires (single/multi/many). t is excluded from $\mathrm{TablePrior}$ because it mainly changes retrieval phrasing (via Q_a) rather than which index to query; including t in routing did not improve dev routing accuracy in our pilot checks.

Feature schema. We use a discrete tuple (h, d, c, t) and its one-hot encoding f, where $h \in \{\text{1-hop}, \text{2-hop}, \text{3-hop}, \text{4-hop+}\}$ (4-hop+ denotes ≥ 4 hops), $d \in \{\text{easy}, \text{medium}, \text{hard}\}$, and $c \in \{\text{single}, \text{multi}, \text{many}\}$. The answer-format cue t is used only for constructing Q_a.

Feature inference. We estimate (h, d, c, t) with an LLM feature extractor (temperature $= 0.0$) that returns structured JSON over the question text only. The extractor outputs reasoning steps, evidence-count class, and retrieval-difficulty class; these fields are mapped to (h, d, c) for $\mathrm{TablePrior}$ and to $\mathrm{BridgeTrigger}$. If parsing fails, we fall back to conservative defaults (`2-hop`, `medium`, `multi`).

LLM routing preference. For each $r \in \mathcal{R}$, an LLM routing prompt (temperature $= 0.0$) returns a normalized score $s_{\text{llm}}(r \mid q)$. The final route score follows Section 3.2: $s(r) = \lambda\, s_{\text{prior}}(r \mid f) + (1 - \lambda)\, s_{\text{llm}}(r \mid q)$ with $\lambda = 0.7$ and margin $\delta = 0.15$.

Manual prior. $\mathrm{TablePrior}(r; h, d, c)$ defines a distribution over $r \in \{\mathrm{CR}, \mathrm{WR}, \mathrm{CR\&WR}\}$ (each row sums to 1). Rows were set on the development split by inspecting route–accuracy patterns: shallow/easy questions receive higher CR mass; deep or hard questions receive higher CR&WR mass; WR-only mass increases when corpus recall is likely insufficient. Table H.1 shows an excerpt of the SAS prior policy.

H.2.1 QAR Retrieval Config

QAR constructs multiple query signals, which are combined into a unified retrieval representation before hybrid dense–sparse retrieval. Table H.2 summarizes the contribution of each signal to this retrieval mixture.

Query signal	Weight	Role
Original (Q_o)	0.5	Primary retrieval cue
Bridge (Q_b)	0.2	Multi-hop bridge cue
Entity-augmented (Q_e)	0.2	Entity cue
Answer-oriented (Q_a)	0.1	Answer-type cue

Table H.2: Query-signal mixture used for unified retrieval representation. Q_b is activated only for multi-hop queries with medium or hard retrieval difficulty.

H.2.2 Unified Retrieval Fusion

QAR combines the active query signals into a unified retrieval representation using the weights in Table H.2. Hybrid dense–sparse retrieval then produces a shared candidate pool using w_d=0.6 and w_b=0.4 for dense and BM25 signals, respectively. Retrieved candidates are normalized into a common schema and assigned rank-based fused scores before reranking. Duplicate candidates are merged by canonical chunk ID while retaining the maximum fused score. WR snippets are converted into the same candidate format and merged through this interface when WR is enabled.

H.2.3 Retrieval Stack

Corpus retrieval uses BGE-small-en-v1.5 (384-d, L2-normalized) with a FAISS IVF index ($nlist$=4096, $nprobe$=32) and a BM25 index (Pyserini/Lucene) over the DPR-processed Wikipedia passage corpus, with 384-token chunks (overlap 48). Candidates in the shared rerank pool are rescored by a lightweight cross-encoder reranker (ms-marco-MiniLM-L6-v2) before TVM selection.

H.2.4 Web Retrieval (WR)

Web retrieval uses the Serper API with top-6 snippets per query, request delay 1.5s, timeout 30s, and up to three retries. Returned snippets are converted into passage-level candidates and merged with corpus candidates in the shared rerank pool. For reproducibility, cached web payloads are stored alongside the corresponding run logs for each CR&WR instance, enabling evaluation replay without re-querying Serper.

H.2.5 Token-Value Maximization (TVM)

Cue set Φ. Given pooled evidence E, we construct Φ from QAR signals and lightweight extractors: (i) Q_o text; (ii) activated Q_b, Q_e, and Q_a strings when present; (iii) answer-type phrases (e.g., person/location/date); and (iv) named entities and salient keywords from the question. Each $\phi \in \Phi$ is embedded with the same encoder used for evidence (BGE-small-en-v1.5). The example below illustrates the resulting cue construction and coverage-based TVM selection under a fixed budget.

> **Question** (Q_o): Which university did the author of *The Old Man and the Sea* attend? ($\mathrm{BridgeTrigger}(f)$=`true`.)
> **QAR**: Q_b = "Who is the author of *The Old Man and the Sea*?"; Q_e augments entity cues for the novel; Q_a targets a person/institution answer.
> **Cue set** Φ: deduplicated $\{Q_o, Q_b, Q_e, Q_a\}$ plus answer-type phrases (e.g., person), entities (*The Old Man and the Sea*), and salient keywords (author, university; $|\Phi| \leq 20$).
> **TVM**: a biography passage e may receive only moderate rerank scores under Q_o alone, but achieves high coverage for cues related to author and university, allowing it to be retained under the fixed token budget.

Coverage $\mathrm{cov}(e, \phi)$**.** For evidence e and cue ϕ, we define a soft coverage score as the cosine similarity between their embeddings (with lexical overlap as fallback when embeddings are unavailable). Greedy TVM uses marginal *soft* coverage gain rather than strict $\{0, 1\}$ indicators; Algorithm 1 in the main text presents the equivalent thresholded view.

Submodularity. The hard coverage objective $f(S)$ in the main text is monotone submodular under a knapsack-style token budget. Greedy marginal-gain-per-token selection therefore follows classical submodular optimization analyses (Nemhauser et al., 1978; Sviridenko, 2004).

Selection procedure. TVM operates on reranked candidates under a compress budget of B_{cmp}=1,600 tokens (80% of the 2,000-token evidence cap). We warm-start with the top-3 reranked items and then greedily add evidence maximizing $\Delta(e)/\mathrm{tokens}(e)$, where $\Delta(e)$ combines (a) newly covered cue mass from Φ and (b) a small rerank-score prior. Selection stops when the token budget is exhausted, no candidate has gain above 0.0003, or early-stop patience (20 steps) is reached.

Token accounting. We track total context usage as

$$\mathrm{Tok}_{\text{total}} = \mathrm{Tok}_{\text{inst}} + \mathrm{Tok}_{\text{evid}},$$

where $\mathrm{Tok}_{\text{evid}}$ counts the fused evidence string passed to the generator and $\mathrm{Tok}_{\text{inst}}$ counts instruction templates; both use the `cl100k_base` tokenizer. **EvTok** in Tables 1–2 reports $\mathrm{Tok}_{\text{evid}}$: the token count of the final fused evidence string passed to the generator (up to the 2,000-token cap). TVM greedy selection runs under B_{cmp}=1,600; optional cluster supplements and route guards may add tokens afterward, so reported EvTok can approach but not exceed the 2,000-token cap (Table F.1).

Deduplication. Before TVM, candidates are deduplicated by passage identifier, document–chunk identifier, or URL-plus-snippet hash; the highest-scoring duplicate is retained (no embedding-based semantic dedup).

H.2.6 Prompt Roles (Release)

We do not reproduce full prompt text in the paper. The released package will include the four prompts used in the main experiments:

- **Feature-extraction prompt** — LLM labeling of (h, d, c) for SAS;
- **Routing prompt** — LLM scoring over $\{\mathrm{CR}, \mathrm{WR}, \mathrm{CR\&WR}\}$;
- **Bridge-query prompt** — optional Q_b generation when the bridge trigger fires;
- **Answer-generation prompt** — final prediction over the fused evidence context.

All prompts use temperature $= 0.0$ for deterministic decoding in the reported runs.

I Reproducibility

We fix random seeds, use the hyperparameters in Table F.1, and follow the dataset pools and evaluation protocol described in this appendix. We will release the TriCo-RAG codebase, unified configurations, evaluation scripts, dataset pools, run logs, prompt templates, and cached web payloads for CR&WR routes.