Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Cost-Aware Gated Retrieval: Decoupling Retrieval Initiation from Evidence-Conditioned Escalation

Zida Hao

20242501431@stu.xju.edu.cn

Xinjiang University

Yubo Li

Xinjiang University

Research Article

Keywords: Retrieval-augmented generation, Runtime scheduling, GPU inference, Uncertainty estimation, KV-cache reuse, Quality-cost tradeoff

Posted Date: November 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11499644/v2

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Cost-Aware Gated Retrieval: Decoupling Retrieval Initiation from Evidence-Conditioned Escalation

Zida Hao[1*] and Yubo Li[1]

[1*]School of Software, Xinjiang University, Urumqi, China.

*Corresponding author(s). E-mail(s): 20242501431@stu.xju.edu.cn;
Contributing authors: 20242501197@stu.xju.edu.cn;

Abstract

Retrieval-augmented generation on a resource-constrained graphics processing unit requires runtime decisions about when to search and escalate. We present Cost-Aware Gated Retrieval (CAGR), separating these decisions through five-token uncertainty signals, evidence-conditioned utility estimates, and a persistent C++/CUDA state machine. Two frozen operating points are evaluated in separate 1,000-question cohorts. The recovery-oriented point restores 0.96 percentage points of HotpotQA F1 over a matched binary gate (95% confidence interval [0.22, 1.83]). The fast-selective point reduces mean latency by 12.77% and retrieval calls by 44.7%, while trading 2.80 HotpotQA F1 points for 3.82 TriviaQA F1 points relative to always retrieving. Independent paired one-sided 95% overhead upper bounds remain below 0.5%. Development-only key-value cache bridging reduces query-generation time by 63.01%; this component result does not establish lossless routed-system quality. The study quantifies runtime savings, recovery, and initiation risk under explicit budgets.

Keywords: Retrieval-augmented generation, Runtime scheduling, GPU inference, Uncertainty estimation, KV-cache reuse, Quality-cost tradeoff

1 Introduction

Retrieval-augmented generation (RAG) combines parametric knowledge with external passages [12, 14], while architectures such as Fusion-in-Decoder and RETRO improve evidence integration [2, 6]. At inference time, question answering requires two decisions: whether to search and whether the retrieved evidence is sufficient. Before searching, the model observes the question and its predictive state; afterward, document scores and a preliminary answer provide additional information about unresolved knowledge needs. Separating these decisions allows computation to follow the evolving evidence, avoiding deeper searches and generations when they leave the answer unchanged.

Prior work offers several ways to allocate retrieval. TARG uses uncertainty from a short no-context draft for a single retrieval decision [23], while Adaptive-RAG selects answering depth from question complexity [8]. SUGAR-L combines semantic uncertainty with multiple retrieval depths and feedback within its iterative branch [29]. Together, these approaches motivate a compact policy that estimates the value of initiating retrieval separately from the value of escalating an evidence-conditioned answer. For resource-constrained inference, realizing this separation also requires efficient access to the model state at each decision boundary.

A routing policy must connect model confidence to the benefit of obtaining evidence, since overconfidence and context sensitivity can weaken confidence-based correctness estimates [17, 20]. We therefore model the difference between alternative action outcomes, distinguishing the gain from a first retrieval, Δ_{01}, from the gain from subsequent retrieval, Δ_{12}. The first target governs access to evidence, whereas the second allocates further computation after evidence has been observed. Both targets retain beneficial, ineffective, and harmful retrieval outcomes.

Selective escalation also dilutes its own dataset-level effect. Against a binary gate with identical non-escalated outputs, the overall gain equals the escalation frequency multiplied by recipients' conditional mean gain. Because the aggregate effect of escalation depends on both selection frequency and recipient-level gain, we report both quantities and account for beneficial, unchanged, and harmful outcomes.

CAGR implements this separation with two small decision rules and a shared inference state. The first rule observes at most five generated tokens. Queries selected for retrieval receive the same preliminary retrieval-augmented answer; only then does the second rule decide whether another search is worth its cost. We implement signal extraction in a persistent native state machine that reuses the greedy selection computation and defers feature conversion until a decision boundary. Queries that skip retrieval resume the existing key-value (KV) cache. Queries that retrieve pay for their discarded prefix and the new evidence-conditioned generation.

We study two complementary operating points. CAGR-R (Recovery-Oriented Operating Point) emphasizes quality recovery through more frequent escalation, whereas CAGR-F (Fast-Selective Operating Point) combines a stronger initiation rule with selective action acceptance. Test Cohort I evaluates CAGR-R and a fixed answer-expansion variant; Test Cohort II evaluates both frozen operating points. Each cohort contains 500 questions per dataset, with the later operating point evaluated on a fresh cohort after development informed by the first cohort. This design supports direct comparisons between configurations within each cohort.

From a systems perspective, the problem is runtime scheduling on a resource-constrained single GPU: retrieval, abandoned-prefix work, query generation, and regeneration consume the latency budget. We examine the scheduler together with the Native State B C++/CUDA reduction and persistent state machine, direct-answer key-value cache reuse, and the development-only KV-Bridge extension. This positioning connects the algorithmic decisions to their execution costs while retaining the measured quality boundaries.

The paper makes four contributions:

1. We formulate retrieval initiation and evidence-conditioned escalation as separate marginal-utility decisions with explicit information boundaries and finite generation and retrieval budgets.
2. We implement five-token routing in a persistent C++/CUDA state machine and measure its incremental overhead through component timing and independent paired on/off execution.
3. We characterize two frozen operating points across two separate test cohorts, including a matched binary comparison and a counterfactual decomposition of missed beneficial searches and avoided harmful searches.
4. We evaluate cache continuation, action alternatives, feature families, prefix truncation, matched-quota random assignment, and retrieval-cost scenarios, distinguishing measured executions from offline diagnostics.

2 Related Work

2.1 Retrieval decisions from model uncertainty

Retrieval and evidence use have developed along several complementary directions. DPR learns a supervised dual-encoder retriever, whereas Contriever learns dense representations through contrastive training [7, 12]. RAG couples retrieval with sequence generation, Fusion-in-Decoder combines multiple retrieved passages, and RETRO conditions language modeling on retrieved chunks [2, 6, 14]. These methods alter retrieval or evidence integration. CAGR instead addresses when to invoke a fixed retrieval interface and when to repeat it after answering.

TARG extracts entropy, a transformed top-two logit gap, or variation across short drafts, then applies a threshold for single-step retrieval [23]. Its accounting for draft cost and prefix-length sensitivity informs the comparisons here. Know Before You Fetch likewise calibrates model signals for graded retrieval budgets and accounts for the initial closed-book computation [5]. Passage-count allocation and retrieval-round allocation expose different information: a second search can respond to an answer generated from the first evidence set.

Uncertainty estimates also depend on what context the model receives. The axiomatic analysis of Soudani et al. questions whether common estimates reliably track correctness after retrieval [20]. SeaKR uses internal states for retrieval and reasoning decisions [25]. Ni et al. study overconfidence and perceived knowledge boundaries [17]. Moskvoretskii et al. compare adaptive retrieval and uncertainty estimators across quality and efficiency measures [16]. Together, these studies motivate model-state signals without equating confidence with retrieval benefit. CAGR fits paired action differences and retains positive, ineffective, and harmful outcomes.

2.2 Choosing depth and updating the search

Adaptive-RAG learns a question-complexity classifier to select retrieval depth [8], whereas SUGAR and SUGAR-L use semantic uncertainty and iterative feedback [28, 29]. CAGR places its escalation decision after a common first retrieved answer, using evidence-conditioned features to estimate the marginal value of another search. Our TARG-style and SUGAR-L-style baselines adapt these routing mechanisms to a shared backbone, corpus, and two-round budget, enabling controlled comparisons of their quality and execution costs.

FLARE triggers retrieval from anticipated low-confidence text [9], while DRAGIN constructs queries from broader context and ETC follows temporal uncertainty trends [15, 21]. IRCoT interleaves retrieval with reasoning, Self-RAG learns reflection tokens for retrieval and critique, and AB-RAG combines confidence and evidence signals under a budget [1, 11, 22]. CAGR pursues lightweight short-answer routing with an unchanged instruction-tuned generator and compact calibrated utility heads.

2.3 Utility, budgets, and execution cost

Budget-aware evaluation distinguishes policy quality from additional spending, including comparisons between offline quotas and deployable thresholds [18]. RetrievalQA examines calibration on questions requiring external knowledge [26]. Our matched-quota and prefix analyses likewise account for the complete computation used to obtain routing signals. Reporting retrieval calls, input/output tokens, and end-to-end latency separately captures the additional work introduced by prefixes, rewriting, and regeneration.

2.4 Cache reuse and inference systems

PagedAttention manages KV memory and sharing in vLLM, SGLang's RadixAttention reuses common prefixes, and FlashAttention reduces attention-memory traffic [4, 13, 27]. Building upon prefix uncertainty signals [23] and KV-cache preservation [13, 27], CAGR introduces a decoupled two-stage marginal-utility formulation. A zero-copy native state machine separates pre-retrieval initiation from post-evidence escalation while preserving useful inference state. KV-Bridge extends this reuse to query generation within an individual question.

3 Methodology

3.1 Actions, information, and marginal utility

Let q denote a question, $\mathcal{D}$ a fixed passage index, and M a fixed generator. Three bounded answer paths produce a direct answer a_0, a first retrieved answer a_1, and an escalated answer a_2. For reference answer set y, let $Q(a, y) \in [0, 1]$ be token F1 and define

$$\Delta_{01} = Q(a_1, y) - Q(a_0, y),$$
$$\Delta_{12} = Q(a_2, y) - Q(a_1, y).$$

Reference answers supply paired action-quality labels during development and final scoring, while routing uses only features available at the current decision boundary. Development observes every fixed path on the same questions,

retaining harmful as well as beneficial action differences. During deployment, the escalation head is evaluated after the first retrieval and preliminary answer have been completed.

Write $R \in \{0,1\}$ for initiation and $E \in \{0,1\}$ for escalation conditional on $R = 1$. The final depth is $T = R(1 + E)$. The available feature vectors are x_0 before retrieval and x_1 after the preliminary answer. Their conditional targets are

$$\mu_{01}(x_0) = \mathbb{E}[\Delta_{01} \mid x_0],$$
$$\mu_{12}(x_1) = \mathbb{E}[\Delta_{12} \mid x_1, R = 1].$$

For complete path costs c_0, c_1, c_2, define $\kappa_{01} = c_1 - c_0$ and $\kappa_{12} = c_2 - c_1$. Lightweight predictors estimate these increments from information available at each boundary. For the frozen CAGR-R implementation, define the nonnegative predicted penalty $\widehat{\kappa}_j^+(x) = \max\{0, \widehat{\kappa}_j(x)\}$, where $j \in \{01, 12\}$. The decision scores are

$$\begin{aligned} s_0(x_0) &= \widehat{\mu}_{01}(x_0) - \lambda_0 \widehat{\kappa}_{01}^+(x_0), \\ s_1(x_1) &= \widehat{\mu}_{12}(x_1) - \lambda_1 \widehat{\kappa}_{12}^+(x_1), \\ R &= \mathbf{1}\{s_0(x_0) > t_0\}, \\ E &= \mathbf{1}\{s_1(x_1) > t_1\}\, \mathbf{1}\{\text{budget permits}\}. \end{aligned}$$

Recorded complete paths supply signed cost-increment fitting targets. Negative observed increments are retained in those records and targets, whereas a negative cost prediction does not provide an online utility bonus. CAGR-R uses $\lambda_0 = 0$ at initiation. Cost weights and thresholds specify the operating preference; Online Resource 2 provides the feature definitions and frozen coefficients for both configurations.

The cost-aware formulation also admits a threshold parameterization. Consider maximizing $\mathbb{E}[E\Delta_{12}]$ subject to $\mathbb{E}[E\kappa_{12}] \le B$ at the second boundary. A resource-price surrogate uses a multiplier $\lambda_1 \ge 0$ to price expected resource use [3]. With an additional acceptance margin t_1, the rule compares predicted gain with $t_1 + \lambda_1 \widehat{\kappa}_{12}^+(x_1)$. If incremental cost is constant, $\widehat{\kappa}_{12}(x_1) = \bar{\kappa}_{12}$, this becomes

$$\widehat{\mu}_{12}(x_1) > t_1', \qquad t_1' = t_1 + \lambda_1 \max(0, \bar{\kappa}_{12}).$$

Under a constant predicted-cost approximation, the clipped resource penalty is absorbed exactly into the acceptance threshold. With variable costs, the omitted term is $\lambda_1[\widehat{\kappa}_{12}^+(x_1) - \max(0, \bar{\kappa}_{12})]$, which can change the selected actions. Explicit cost penalties model this variation directly, whereas a fixed gain threshold provides a compact resource preference under a common-cost approximation. Hard execution budgets bound realized work in either parameterization.

Figure 1 summarizes the two information boundaries and complete execution paths.

3.2 Five-token signals and evidence-conditioned features

At decoding position t, let $z_t^{(1)}$ and $z_t^{(2)}$ be the largest and second-largest raw logits. With $m = \min(K, L)$ observed positions, where L is the generated length before termination, we compute

$$g_t = z_t^{(1)} - z_t^{(2)}, \quad \bar{g} = \frac{1}{m} \sum_{t=1}^{m} g_t,$$

$$u = \frac{1}{m} \sum_{t=1}^{m} \exp(-g_t/3), \qquad K = 5.$$

We also use $v = \exp(-\bar{g}/3)$ in the recovery-oriented utility head, preserving the distinction between transforming a mean and averaging transformed gaps. The observation window includes early end-of-sequence positions and ends at the actual generated length. All signals use raw logits, with top-two logit gaps measured before softmax normalization.

CAGR-R uses $(\bar{g}, v, n_q, n_c)$ for initiation, where n_q counts bounded question terms and n_c counts capitalized non-stopword terms. Its escalation head uses the preliminary answer's mean gap, top-ten retrieval-score variance, and generated answer length, excluding tokens in the termination-token set. Complete output-token cost accounting separately includes every generated position, including a generated termination token. Score variance defaults to zero when fewer than two candidates are available. Standardized linear ridge models predict gain and cost, with standardization fitted inside each training fold and deployment heads fitted on the complete Development Calibration Split.

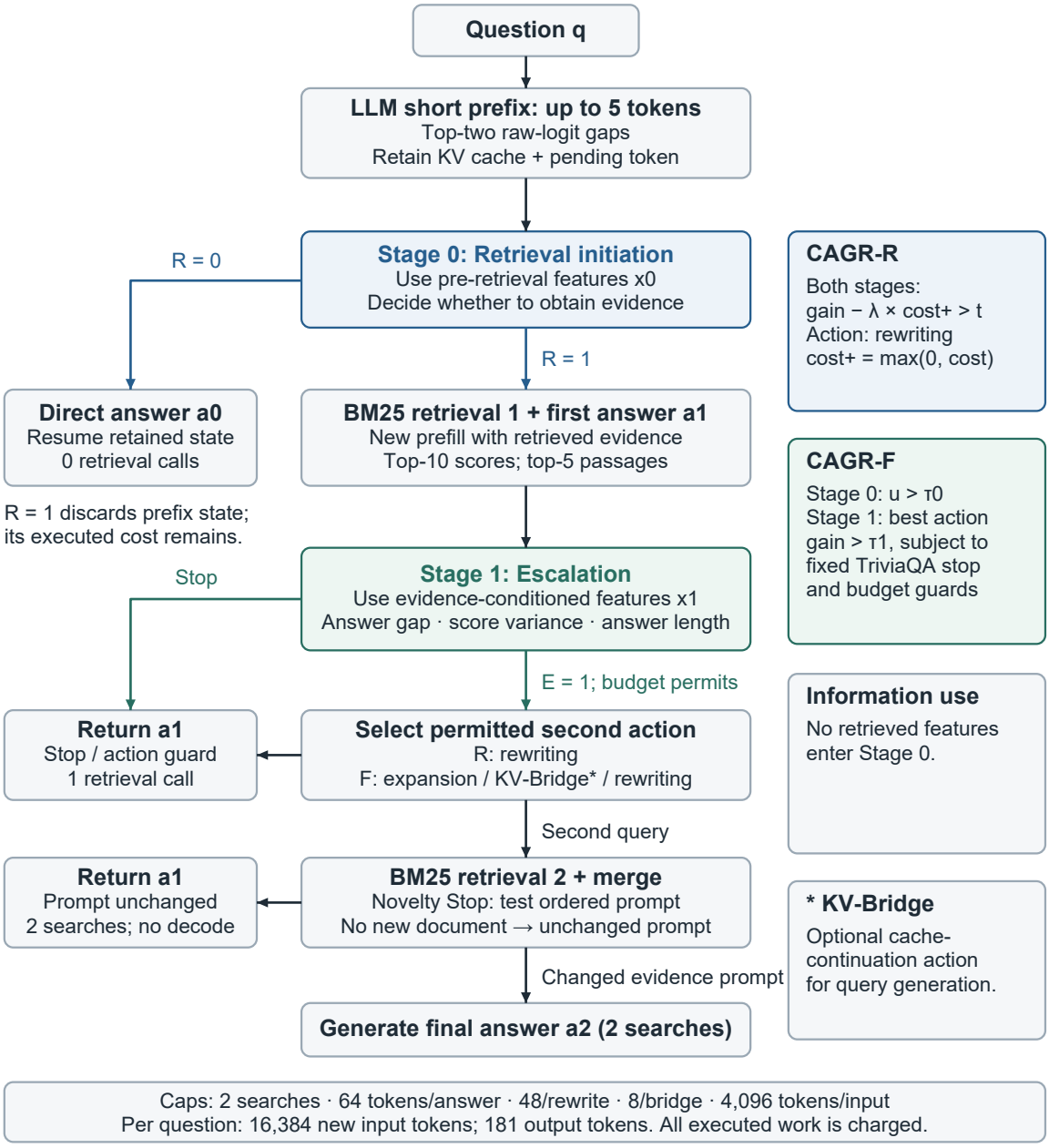

Fig. 1 Two-stage cost-aware retrieval. The blue gate decides whether to initiate retrieval from a prefix of at most five tokens. Skipped questions resume the retained state; retrieved questions generate a common first answer before the green evidence-conditioned gate acts. CAGR-R uses utility and cost predictions; CAGR-F uses a prefix threshold and action-specific gain predictions with fixed guards. A second search can return the first answer when the evidence prompt is unchanged. Both searches and any query generation remain charged. Caps bound all paths.

CAGR-F thresholds u for initiation and scores three escalation actions using the same evidence-conditioned features, accepting the highest predicted gain above a stricter threshold. This directly calibrated gain threshold expresses the common-cost resource preference described above. A fixed dataset-conditioned rule suppresses high-confidence TriviaQA escalation. The two operating points thus share a five-token signal while using distinct initiation and action rules.

3.3 Bounded escalation and Novelty Stop

Every selected question first receives one BM25 retrieval and an answer from the common evidence template. The main escalation action asks the same generator for one short search query using the question, preliminary answer, and retrieved evidence. The rewritten query can introduce a bridge entity absent from the question. A second BM25 retrieval is merged with the first context using document identifiers and a fixed order. The generator then answers from the updated context. Query rewriting is a generation call with its own prefill and decoding cost.

Novelty Stop avoids a final generation when the second retrieval adds no document and preserves the ordered evidence prompt. For prompt P and deterministic decoding F, the equality $P_2 = P_1$ gives $F(P_2) = F(P_1) = a_1$, allowing the controller to return a_1. The resulting saving is the omitted final generation.

All paths permit at most two retrieval calls. Answer generations are capped at 64 tokens, query rewriting at 48 tokens, and each input at 4,096 tokens. The shared per-question limits are 16,384 newly processed input tokens and 181 output tokens. The controller checks the remaining budget before an action and reserves capacity for the final answer. These limits make escalation finite even if the preliminary answer remains uncertain.

3.4 Persistent native state and direct-answer KV reuse

CAGR combines greedy token selection and top-two signal extraction in a shared reduction over the FP16 vocabulary logits. Persistent buffers hold the five-position observation window, partial reductions, token indices, validation flags, and fixed host records. Each observed position uses one bound native operation with two device passes. A lowest-index tie rule determines the selected token, while validation rejects nonfinite logits.

The state machine accumulates observations in persistent records and converts them to bounded features at the decision boundary. Greedy selection and signal extraction share the native operation, avoiding additional feature-specific synchronization and per-position timing events. End-to-end latency includes ordinary host–device interaction, with feature time measured as a component of the generation path.

The direct branch pauses after the prefix with its KV cache and pending token intact. When $R = 0$, decoding resumes from this state after a single prefill of the question prompt. When $R = 1$, the evidence-conditioned prompt receives a new prefill and the complete cost includes the abandoned prefix. This branch-specific accounting captures both state reuse and regeneration.

3.5 KV-Bridge as a cache continuation extension

KV-Bridge retains the first evidence-conditioned answer state and appends a short instruction requesting bridge terms. The cache contains the processed context and all answer tokens except the current pending token. Continuation feeds that pending token exactly once, followed by the new instruction suffix. Only the suffix and subsequent output positions require new processing; the evidence documents are not prefilled again. Attention positions and cache length advance from the retained state.

The extension caps bridge output at eight tokens and combines the generated terms with the question, answer, and evidence clues for the second retrieval. Empty terms, pure question repetition, or a truncated nonterminated bridge can trigger stopping after the bridge generation. An identical-prefix replay control checks token equality and cache transitions, while a separate comparison measures the complete short bridge instruction against the longer rewriting task.

KV-Bridge provides a cache-continuation extension evaluated in the development studies.

Test Cohort II characterizes CAGR-F's initiation and selective-escalation configuration, which selected the other available actions.

Figure 2 details branch-specific state reuse and the optional cache-continuation extension.

4 Experimental Setup

4.1 Backbone, index, and execution environment

All experiments used an isolated Linux evaluation server with NVIDIA Tesla T4 GPUs (16 GB GDDR6 VRAM), pinning the model to a single `cuda:0` device. Batch size and concurrency were both 1. The software environment comprised Python 3.13.15, PyTorch 2.11.0 (CUDA 12.8), and HuggingFace Transformers 4.51.3 in FP16 precision. The generator was Qwen2.5-3B-Instruct [19], with model and tokenizer fixed to revision `aa8e72537993ba99e69dfaafa59ed015b17504d1`. Answer decoding used greedy selection with scaled dot-product attention.

The small backbone and locally staged index define a controlled single-device benchmark for resource-constrained inference. Across the completed cohorts, 54,571 generation calls cover normal methods, technical measurements, warm-up, and cache checks, including 8,000 technical on/off executions in 4,000 pairs. Holding the backbone and index fixed isolates routing-mechanism comparisons under shared retrieval conditions.

The 483,690-passage SQLite FTS5 index was staged onto local working storage before inference to stabilize retrieval I/O. It contains deduplicated passages from HotpotQA training contexts and serves both datasets and all methods. FTS5 uses Porter stemming with Unicode tokenization, title/text weights of 2:1, and at most 24 query terms. Each retrieval round returns ten candidates for score statistics and five passages of at most 256 tokens for the evidence prompt.

Index staging, integrity checks, model loading, native compilation, and warm-up are recorded as preparation costs separately from steady-state question latency. Development feature replays and figure preparation use the saved experimental records.

OpenAI Codex was used to assist with experimental code development and manuscript preparation. The authors verified the implementation and results and take responsibility for the final manuscript.

4.2 Development and separate test cohorts

The Development Calibration Split contains 200 questions, with 100 per dataset. Five balanced folds provide out-of-fold (OOF) predictions using 160 training questions per fold and fold-specific preprocessing. Deployment heads are subsequently fitted on all 200 development questions. Each test cohort contains 1,000 public validation questions, balanced between HotpotQA and TriviaQA and held out specifically for this study [10, 24].

Question identifiers and normalized near-duplicate checks separate development, technical checks, and both test cohorts. The initial cohort retains its original fixed sample draw, while subsequent inference and bootstrap analyses use seed 42.

For each test cohort, inference uses question-only inputs, with reference answers supplied separately to the scorer after predictions are complete. Methods, thresholds, prompts, budgets, and model weights are fixed before scoring the relevant cohort. CAGR-R was frozen before Test Cohort I; CAGR-F was developed on the same development split after Cohort I analysis and frozen before the fresh Test Cohort II. We report the cohorts separately and identify analyses of previously examined outputs as post hoc diagnostics.

4.3 Eight-method comparisons and fair cost boundaries

Direct produces a closed-book answer. Always-RAG retrieves once for every question. Always-Multihop always attempts the fixed two-round procedure, subject to Novelty Stop. Matched Binary shares the corresponding CAGR initiation rule, prefix, and first-answer path, but never escalates. It isolates the observed effect of adding the second decision and its chosen actions. TARG-style uses the transformed prefix-gap threshold under the present backbone and a five-token prefix; its original paper uses a longer default prefix. The baseline threshold is calibrated on development data and fixed within each cohort.

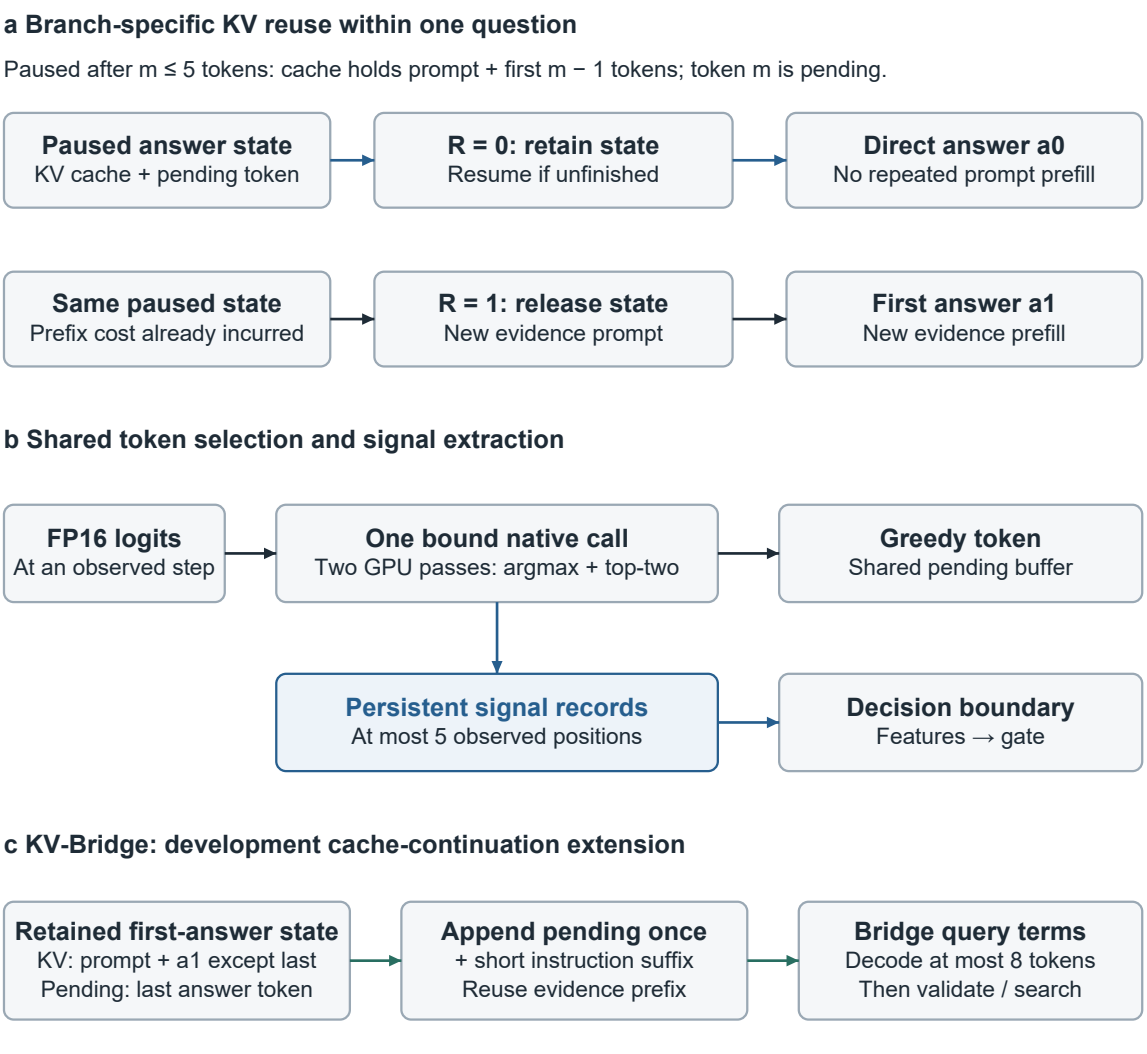

Fig. 2 State reuse and shared native signal extraction. (a) The direct branch retains its cache and pending token, while the retrieval branch requires a new evidence prefill. (b) One bound native operation shares greedy selection and top-two extraction through two GPU passes; persistent records are converted to features at the decision boundary. (c) The optional development KV-Bridge feeds the pending answer token once and appends a short instruction, reusing the evidence prefix. Its complete call changes the query-generation task; the separate identical-prefix control isolates cache savings.

SUGAR-L-style adapts semantic-feedback routing to the shared backbone and two-round budget. It uses two stochastic answer samples of at most eight tokens and bidirectional entailment checks of at most four tokens, reassessing semantic uncertainty after retrieval. Its complete cost includes sampled answers and entailment calls. The shared candidate count, retrieval budget, and compression choices define the controlled adaptation used here. Test Cohort I includes CAGR-R and its fixed answer-expansion variant, while Test Cohort II includes both frozen operating points, giving eight methods per table.

Every question's methods execute on the same device within its session, with a rotating counterbalanced order. Each cohort is evaluated in two balanced sessions of 500 questions, each containing 250 questions per dataset. Cross-method comparisons are paired within questions; session is retained as a bootstrap stratum. End-to-end wall time includes all work inside the method boundary: observed and discarded prefixes, retrievals, context processing, query generation, feedback generation, final answers, and host decisions. Technical on/off executions and warm-up calls are recorded separately from normal-method means.

4.4 Quality metrics, bootstrap estimators, and overhead

We report normalized exact match (EM) and token F1, taking the best match across the saved reference aliases. Normalization lowercases text, removes punctuation and articles, and standardizes whitespace. HotpotQA yes/no/no-answer mismatches follow its answer-scoring conventions. Overall results give the two datasets equal weight because each contributes 500 questions. HotpotQA supporting-fact and joint metrics are outside this short-answer evaluation.

For metric m and methods A, B, the paired estimator within a cohort is

$$\widehat{d}_{A-B} = \sum_s w_s \frac{1}{n_s} \sum_{i \in s} (m_i^A - m_i^B), \quad w_s = n_s/N,$$

where s indexes dataset-by-session strata, restricted to the selected dataset for a dataset-specific comparison. We resample questions with replacement inside each stratum, preserving method pairs, for 20,000 replicates. The 2.5th and 97.5th percentiles give marginal two-sided 95% intervals. Speed ratios are ratios of resampled mean latencies, not means of per-question ratios. These intervals are not adjusted for all reported comparisons.

Development calibration used a reference of 1.0 percentage point for the allowable sample-mean EM and F1 decrease relative to Always-RAG in each dataset. Test analysis reports the resulting quality–latency tradeoff through paired point estimates and marginal confidence intervals. This calibration reference is a practical operating target rather than a statistical noninferiority margin.

We measure routing overhead in two ways. Component overhead is the mean of each question's feature-and-decision time divided by its full path time. Independent overhead uses two balanced on/off pairs per question with the scientific path held fixed. For pair j, let $r_{ij} = (\ell_{ij}^{\text{on}} - \ell_{ij}^{\text{off}})/\ell_{ij}^{\text{off}}$ and $\bar{r}_i = (r_{i1} + r_{i2})/2$. The estimator averages $\bar{r}_i$ across questions; the two pairs are not treated as independent questions. Negative timing differences and extreme observations are retained. The 95th bootstrap percentile is the one-sided upper bound, compared with 0.5%. Cohort I retains its originally specified unstratified within-dataset question bootstrap for this overhead estimator; Cohort II uses dataset-by-session strata. Cross-method quality and cost comparisons use the stratified paired design in both cohorts.

5 Empirical Results on Test Cohort I and Test Cohort II

5.1 Complete main comparisons

CAGR-R and CAGR-F occupy different parts of the measured quality-cost space. Tables 1 and 2 report each cohort separately and retain the stronger, more expensive baselines. Overall quality is insufficient by itself: the two datasets respond differently to skipped retrieval, so each dataset's EM and F1 accompany the equal-weight average.

In Test Cohort I, CAGR-R makes 0.501 retrieval calls per question, a 49.9% reduction from Always-RAG. Mean latency is 592.31 versus 618.33 ms, with a paired difference of -26.02 ms (95% CI $[-61.22, 9.44]$ ms). The two-round and

Table 1 Test Cohort I: eight methods on 500 HotpotQA and 500 TriviaQA questions. EM and F1 are percentages.

Method	H EM	H F1	T EM	T F1	All EM	All F1
Direct	14.40	21.13	30.40	38.63	22.40	29.88
Always-RAG	24.40	33.59	35.20	41.38	29.80	37.48
Always-Multihop	26.80	35.92	35.20	41.73	31.00	38.83
Matched Binary	21.40	30.65	35.60	43.06	28.50	36.85
TARG-style	22.40	32.02	37.20	44.40	29.80	38.21
SUGAR-L-style	23.60	33.49	39.40	44.84	31.50	39.16
CAGR-R	22.00	31.60	35.40	42.80	28.70	37.20
CAGR-R (answer expansion)	21.40	30.72	35.80	43.13	28.60	36.93

Method	Latency (ms)	Retr./q
Direct	209.43	0.000
Always-RAG	618.33	1.000
Always-Multihop	1916.15	2.000
Matched Binary	473.82	0.410
TARG-style	549.46	0.555
SUGAR-L-style	2460.22	1.240
CAGR-R	592.31	0.501
CAGR-R (answer expansion)	516.69	0.501

semantic-feedback adaptations take 3.24 and 4.15 times as long on average, respectively. SUGAR-L-style attains higher overall F1, illustrating the quality–cost tradeoff among the measured policies.

Figure 3 summarizes these measurements.

Within Cohort II, CAGR-R and CAGR-F express complementary recovery and speed preferences. CAGR-R reaches 38.07% overall F1 at 540.90 ms, whereas CAGR-F reaches 37.76% at 491.23 ms. CAGR-R's F1 point estimate exceeds Always-RAG's 37.25% and TARG-style's 37.73%, while its latency is below Always-RAG's 563.13 ms. Its 0.516 retrievals per question also compare favorably with TARG-style's 0.561, reflecting the allocation achieved by its combined initiation and action rules.

CAGR-F reaches 29.70% overall EM, compared with 29.10% for Always-RAG and 29.70% for TARG-style. HotpotQA F1 is 35.48%, versus TARG-style's 35.10%, while TriviaQA F1 is lower. The overall F1 difference from TARG-style is 0.0308 points (95% CI $[-0.3775, 0.4317]$), placing their observed quality close together. Section 5.3 examines the escalation frequency and recipient-level gains underlying these aggregate differences.

Figure 4 summarizes these measurements.

5.2 Near-zero incremental routing overhead

Routing adds a small fraction of full execution time in the evaluated configuration. Component means and one-sided upper bounds remain below 0.18% across datasets and cohorts, while independent on/off upper bounds remain below 0.5%. These measurements isolate feature extraction and decision computation within complete paths whose costs include prefix generation, query rewriting, and answer generation.

Table 3 gives both measurement scopes.

For TriviaQA in Test Cohort I, the independent one-sided 95% upper bound is 0.48198%, compared with an upper endpoint of approximately 0.5132% for the two-sided interval. The overhead criterion uses the one-sided bound and includes every question with both fixed on/off pairs.

The corresponding visual summary is provided in Online Resource 1 (Fig. S1).

5.3 Recovery-oriented and fast-selective behavior

The matched binary comparison separates escalation value from its frequency. Let $A_i = R_i E_i$ indicate an executed escalation and $p_{\text{esc}} = n_{\text{esc}}/N$

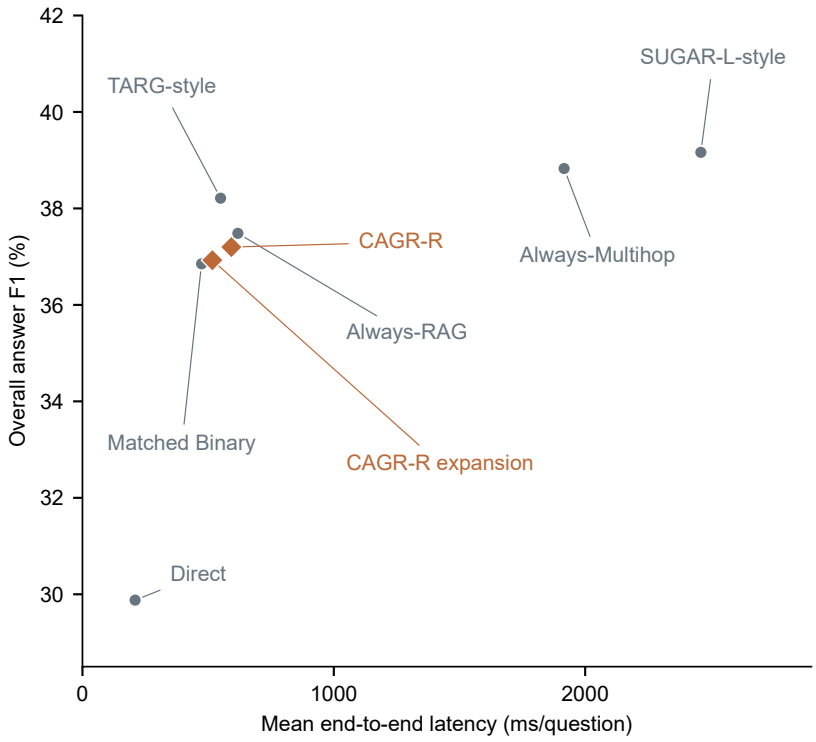

Fig. 3 Test Cohort I quality and latency. Points show means over the same 1,000 questions; paired confidence intervals are reported in the text and supplementary tables

use the full dataset denominator. Because non-escalated outputs match the binary control, the finite-sample identity is

$$\begin{aligned}\Delta\mathrm{F1}_{\mathrm{dataset}} &= \frac{1}{N}\sum_i A_i \Delta_{12,i} \\ &= p_{\mathrm{esc}}\,\overline{\Delta}_{12|A=1}.\end{aligned}$$

The identity expresses aggregate recovery as the product of escalation frequency and conditional mean gain among recipients. Reporting both terms reveals the contribution of additional computation while retaining positive, unchanged, and harmful answer outcomes.

In Cohort I, CAGR-R escalates 63 of 500 HotpotQA questions, or 12.6%, with a conditional mean F1 gain of 7.5826 percentage points. Their product gives a dataset contribution of 0.9554 points (paired 95% CI $[0.2156, 1.8335]$). Nine escalations improve F1, 52 tie, and two reduce it. Relative to Always-RAG, the resulting HotpotQA F1 and EM are lower by 1.9854 and 2.40 points, respectively.

CAGR-R takes 590 direct, 319 single-retrieval, and 91 escalated paths in Cohort I. Its recovery-oriented allocation has a P95 latency of 2,094.98 ms, versus 1,144.52 ms for Always-RAG. Mean input tokens fall from 637.913 to 470.633, while output tokens rise from 6.487 to 9.334 as prefixes and escalation generations contribute to complete execution.

CAGR-F takes 457 direct, 533 single-retrieval, and ten escalated paths in Cohort II. On HotpotQA, seven escalations cover 1.4% of questions and yield a conditional mean F1 gain of 22.1429 points, contributing 0.3100 points across the dataset. Two improve F1, four tie, and one reduces it. Across both datasets, the 1% escalation rate produces a 0.0683-point overall increment over Matched Binary. This small aggregate effect combines sparse execution with positive recipient-level recovery; the HotpotQA increment interval spans zero.

CAGR-F's mean latency is 491.23 ms, 12.77% below Always-RAG, with a paired difference of -71.90 ms (95% CI $[-95.53, -47.91]$ ms).

Table 2 Test Cohort II: eight methods on 500 HotpotQA and 500 TriviaQA questions. EM and F1 are percentages.

Method	H EM	H F1	T EM	T F1	All EM	All F1
Direct	17.40	24.71	31.60	38.60	24.50	31.66
Always-RAG	28.20	38.27	30.00	36.22	29.10	37.25
Always-Multihop	29.20	40.12	29.60	36.18	29.40	38.15
Matched Binary	25.80	35.17	33.60	40.22	29.70	37.69
TARG-style	25.80	35.10	33.60	40.36	29.70	37.73
SUGAR-L-style	28.40	38.35	36.40	42.43	32.40	40.39
CAGR-R	24.40	33.84	35.00	42.31	29.70	38.07
CAGR-F	26.00	35.48	33.40	40.04	29.70	37.76

Method	Latency (ms)	Retr./q
Direct	190.71	0.000
Always-RAG	563.13	1.000
Always-Multihop	1810.69	2.000
Matched Binary	476.04	0.543
TARG-style	486.09	0.561
SUGAR-L-style	2354.80	1.255
CAGR-R	540.90	0.516
CAGR-F	491.23	0.553

Table 3 Routing overhead for CAGR-R in Test Cohort I and CAGR-F in Test Cohort II; N=500 questions per row. U95 is a one-sided 95% upper bound.

Cohort	Dataset	Component mean (%)	Component U95 (%)	On/off mean (%)	On/off U95 (%)
I	HotpotQA	0.14551	0.15107	0.16032	0.29495
I	TriviaQA	0.17365	0.17899	0.31343	0.48198
II	HotpotQA	0.12514	0.12847	0.28488	0.42126
II	TriviaQA	0.13119	0.13506	0.07234	0.20152

Retrieval calls fall from 1,000 to 553. Compared with CAGR-R in the same cohort, CAGR-F is 49.66 ms faster with 0.3130 points lower overall F1. CAGR-R escalates 97 questions versus CAGR-F's ten while initiating fewer first searches, demonstrating how query rewriting and regeneration shape latency alongside retrieval counts. Across their respective primary cohorts, the operating points trade 1.99–2.80 F1 points on multi-hop questions for a 44.7%–49.9% reduction in retrieval calls, with CAGR-F delivering the 12.77% end-to-end speedup. These reductions coexist with TriviaQA F1 improvements of 1.42 points for CAGR-R in Cohort I and 3.82 points for CAGR-F in Cohort II relative to Always-RAG. The matched-binary comparison above estimates recovery under the same initiation rule; it does not remove the cost of missed first retrievals.

TARG-style is a close cost comparator at 486.09 ms, 5.14 ms below CAGR-F; the paired interval for CAGR-F minus TARG-style is $[-6.07, 17.95]$ ms. CAGR-F's HotpotQA and overall F1 point estimates are higher, with intervals that include zero. Relative to the more expensive adaptations, CAGR-F is 3.69 times faster than Always-Multihop and 4.79 times faster than SUGAR-L-style in mean latency, with their corresponding quality values shown in the main table.

The mean savings coexist with a heavier tail than single-pass retrieval. In Cohort II, P95 is 920.64 ms for Always-RAG, 992.22 ms for Matched Binary, and 1,002.26 ms for TARG-style. Prefix-gated systems combine short direct paths with retrieved paths that also incur an abandoned prefix, an execution pattern consistent with this ordering. CAGR-F's P95 is 1,027.15 ms, or 24.90

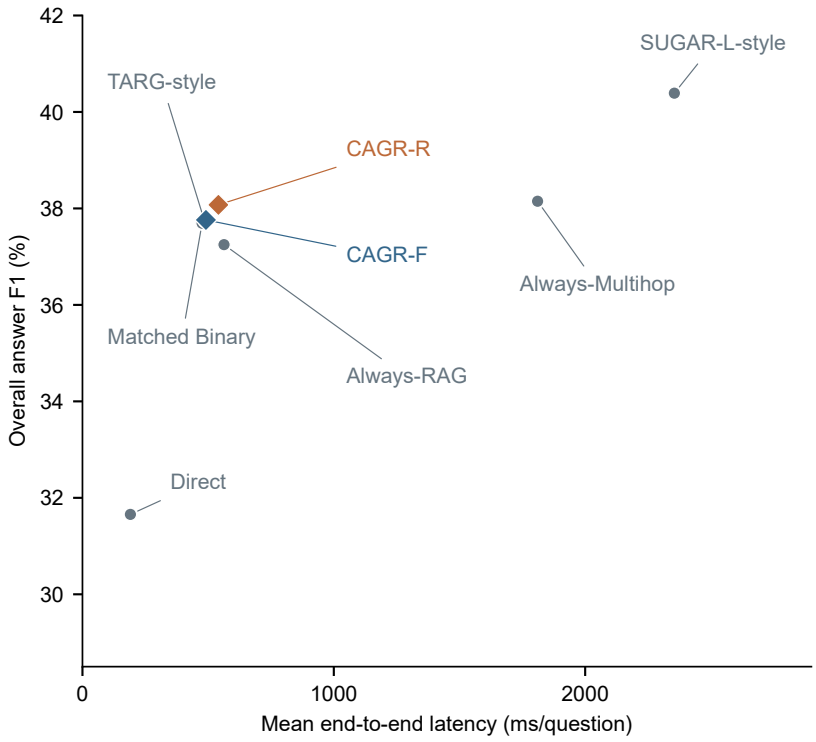

Fig. 4 Test Cohort II quality and latency. Points show means over the same 1,000 questions for both frozen operating points and the comparison methods

ms (2.48%) above TARG-style using unrounded values. It is 61.7% below Always-Multihop's 2,684.22 ms and 75.6% below SUGAR-L-style's 4,215.52 ms. Online Resource 1 (Table S8) gives the complete descriptive latency summaries.

5.4 Asymmetric initiation risk across datasets

The main quality deficit appears before the escalation decision. Because Direct and Always-RAG answers were saved for every question, we can decompose the first gate's observed quality exactly. For skipped set $S = \{i : R_i = 0\}$ in a dataset of size N_d, define

$$H = \frac{1}{N_d} \sum_{i \in S} \max(-\Delta_{01,i}, 0),$$

$$M = \frac{1}{N_d} \sum_{i \in S} \max(\Delta_{01,i}, 0).$$

Then the matched binary gate's F1 difference from Always-RAG is $H - M$. Here H is benefit from avoiding harmful retrieval, and M is loss from missing beneficial retrieval. The denominator is all 500 questions, not only the skipped subset.

In Test Cohort II, CAGR-F skips 209 HotpotQA questions. It misses 46 beneficial retrievals, losing 6.9176 dataset F1 points, and avoids 25 harmful retrievals, gaining 3.8124 points. The net first-stage effect is −3.1052 points. Seven escalations recover 0.3100 points, leaving the final −2.7952-point gap. On TriviaQA, the gate skips 248 questions, misses 39 beneficial retrievals at a cost of 5.5328 points, and avoids 68 harmful retrievals for a gain of 9.5261 points. The resulting first-stage effect is +3.9933 points; three escalations subtract 0.1733 points.

Figure 5 summarizes these measurements.

The asymmetry reveals different information needs across the two datasets. A confident short prefix can conceal a missing relational bridge in a multi-hop question, while noisy passages can

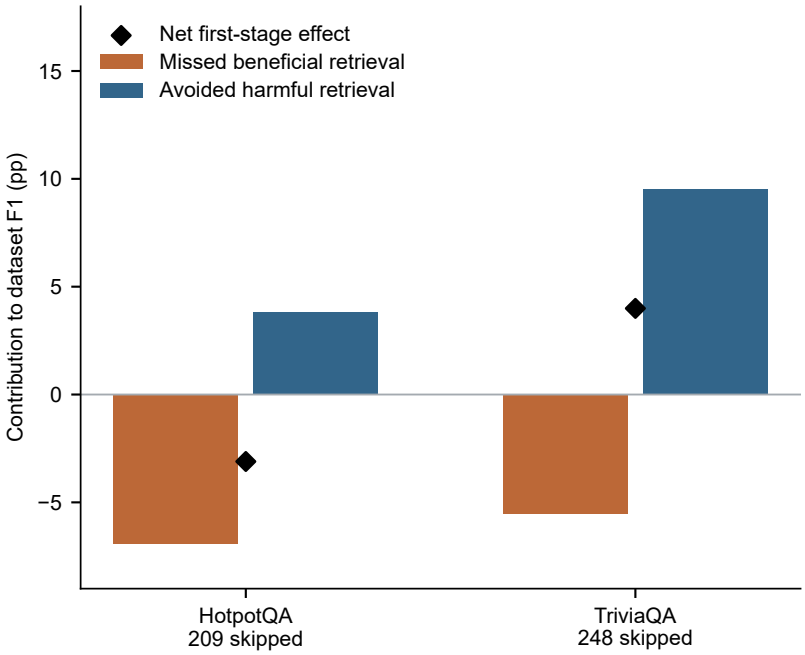

Fig. 5 First-retrieval risk in Test Cohort II. Bars show missed gains and avoided harm among skipped questions, scaled by the full dataset size of 500. Diamonds mark the net first-stage effect

disrupt a factoid answer already represented in model knowledge. TARG-style retrieves only eight additional HotpotQA questions and ten additional TriviaQA questions, indicating that question selection contributes alongside the overall retrieval allocation.

In the same Cohort II comparison, TARG-style's HotpotQA F1 is 3.17 points below Always-RAG, alongside CAGR-F's 2.80-point deficit. The decomposition above attributes a net 3.11-point loss to CAGR-F's first decision, including 46 missed beneficial searches; escalation then provides partial recovery. This shared pattern supports interpreting the observed multi-hop tradeoff as initiation risk in the tested prefix-gated systems, rather than attributing it solely to escalation or cache implementation. It does not establish a universal limit for all retrieval gates.

Cohort I further separates initiation allocation from escalation recovery. On HotpotQA, TARG-style exceeds the matched binary first stage by 1.3750 F1 points, of which CAGR-R escalation recovers 0.9554 points, leaving a final gap of 0.4195 points. The saved-answer quota diagnostic in Online Resource 1 (Sect. S4.1) recovers 0.9011 points by matching dataset-specific first-search counts while preserving the utility ranking. The remaining 0.4738-point first-stage gap identifies an additional contribution from question selection.

6 System extensions and development diagnostics

Online Resource 1 (Sect. S1) reports the full development studies, with source values in Online Resource 2. Across 200 paired development questions, cache-continuing KV-Bridge reduces mean query-generation time from 724.52 to 267.98 ms, a 63.01% reduction (a 2.70-fold call-level speedup). Full rewriting and the eight-token bridge produce gain/tie/harm counts of 17/172/11 and 9/179/12, respectively, relative to the first retrieved answer. These results characterize a tradeoff between query-generation cost and answer recovery. A

stricter 20-question identical-prefix control isolates cache reuse, saving 190.63 ms while preserving outputs.

On the 200-question development split, the five-token signal achieves an out-of-fold area under the receiver operating characteristic curve (AUC) of 0.6615 for positive first-retrieval gain. Thirty question-structure features yield 0.4985 and their fusion with the signal yields 0.6287, favoring compact model-state information in this comparison. Adding evidence-text features to the escalation head raises AUC from 0.6467 to 0.6741, with a paired difference interval of [-0.0341, 0.0935], while routed quality falls at common thresholds.

Matched-quota random assignments support selection value from the saved signals. Truncating saved prefixes from five tokens to one changes CAGR-R's first-search fraction from 41.0% to 81.2% under the fixed head, showing the interaction between observation length and calibration. Novelty Stop omits 11 final generations in Cohort I and two in Cohort II while retaining preceding costs. Fixed-path retrieval-price scenarios quantify sensitivity to retrieval cost with answer quality and routing held at their measured values. These measurements separate three runtime functions: the scheduler selects paid actions, Native State B makes routing state inexpensive to observe, and cache continuation reduces one query-generation component. Neither low routing overhead nor a cheaper query establishes lossless policy quality. The engineering result is a measured latency–quality tradeoff under the stated single-GPU workload and fixed budgets, with TARG-style retained as a competitive control.

7 Limitations and Conclusion

7.1 Limitations

First-retrieval recall remains the principal quality limitation on multi-hop questions. A confident prefix can omit relational evidence, and escalation can act only after an initial retrieval. Both operating points exceed the development reference of a 1.0-percentage-point HotpotQA EM and F1 decrease relative to Always-RAG. Escalation is sparse: CAGR-F selects ten questions in Cohort II, with two improvements, six ties, and two harms, limiting action-specific reliability estimates. Repeated design choices on the 200-question development split also limit how precisely the development feature comparisons generalize.

Evaluation covers two public QA benchmarks, Qwen2.5-3B-Instruct, serial FP16 inference on one T4, and a locally staged FTS5 index of 483,690 HotpotQA training passages. This index has limited coverage for TriviaQA. Larger backbones, dense retrievers, full Wikipedia corpora, and concurrent batched serving require implementation checks, recalibration, and fresh system-overhead measurements. CAGR-F also uses dataset identity in its high-confidence escalation stop; deployment on unidentified task mixtures requires recalibration or a task-independent stopping rule.

CAGR-R and CAGR-F represent two frozen, discrete operating points with different initiation and action rules. Their primary cohorts come from separate execution sessions, so quality and speed comparisons are paired within each cohort. TARG-style and SUGAR-L-style are controlled mechanism adaptations under a shared backbone and budget; the comparisons apply to these implementations. The strong binary gate remains competitive in quality and latency. The finite comparisons support specific resource-constrained quality–cost tradeoffs rather than a global Pareto frontier across configurations.

7.2 Conclusion

CAGR decouples retrieval initiation from evidence-conditioned escalation through marginal-utility routing and a persistent native inference state machine. Across two frozen 1,000-question test cohorts, independent paired measurements place the one-sided 95% upper bound on incremental routing overhead below 0.5%. The cohorts demonstrate two complementary operating points for cost-aware retrieval. In Test Cohort I, CAGR-R reduces retrieval calls by 49.9% relative to Always-RAG and recovers 0.96 percentage points of HotpotQA token F1 over a matched binary gate. In Test Cohort II, CAGR-F reduces mean end-to-end latency by 12.77% (from 563.13 to 491.23 ms) and retrieval calls by 44.7% relative to Always-RAG. This operating point trades 2.80 percentage points of HotpotQA F1 for faster execution, with higher overall EM

and F1 point estimates. Counterfactual decompositions identify first-retrieval recall on multi-hop questions as the primary target for future quality improvements.

Statements and Declarations

Funding

No funding was received for conducting this study.

Competing interests

The authors have no relevant financial or non-financial interests to disclose.

Author contributions

Zida Hao: Conceptualization, methodology, software, investigation, formal analysis, and writing–original draft. Yubo Li: Writing–review and editing.

Ethics approval and consent to participate

Not applicable. This study evaluates computational methods on existing public question-answering benchmarks and involves no recruitment of human participants or animal experiments.

Consent for publication

Not applicable.

Data availability

The study uses HotpotQA and TriviaQA from their original sources [10, 24]. Online Resource 2 provides aggregate source-data tables, question-level quality and cost measurements, paired timing observations, and frozen configuration exports. Benchmark text, model weights, and source passages are obtained from their original providers under the applicable terms.

Code availability

The reproducibility code is publicly available at https://github.com/human-com-h/CAGR. Online Resource 3 provides the frozen scientific implementation and configuration sources for the two test cohorts, together with dependencies and reproduction instructions.

Preprint

An earlier version is publicly available on Research Square as version 1 (https://doi.org/10.21203/rs.3.rs-11499644/v1). The submission retains revised references and method schematics, with journal-specific formatting and systems-oriented framing.

Supplementary information

Online Resource 1 (ESM_1.pdf): Extended development studies, frozen decision parameters and algorithm, complete cost and paired-comparison tables, prefix and quota diagnostics, and reproducibility details.

Online Resource 2 (ESM_2.zip): Fifteen aggregate source-data tables, per-question test quality and cost values, paired timing observations, and frozen routing and generation configuration exports, with a data dictionary.

Online Resource 3 (ESM_3.zip): Frozen scientific code and configuration sources for the two test cohorts, with dependency and scope notes.

References

[1] Asai A, Wu Z, Wang Y, et al (2024) Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In: The Twelfth International Conference on Learning Representations, URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/25f7be9694d7b32d5cc670927b8091e1-Abstract-Conference.html

[2] Borgeaud S, Mensch A, Hoffmann J, et al (2022) Improving Language Models by Retrieving from Trillions of Tokens. In: Proceedings of the 39th International Conference on Machine Learning, pp 2206–2240, URL https://proceedings.mlr.press/v162/borgeaud22a.html

[3] Boyd S, Vandenberghe L (2004) Convex Optimization. Cambridge University Press, https://doi.org/10.1017/CBO9780511804441

[4] Dao T, Fu D, Ermon S, et al (2022) FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In: Advances in Neural Information Processing Systems,

pp 16344–16359, https://doi.org/10.52202/068431-1189

[5] Dong Z, Qin F, Shah M, et al (2026) Know Before You Fetch: Calibrated Retrieval-Budget Allocation for Retrieval-Augmented Generation. arXiv preprint arXiv:2606.29959, URL https://arxiv.org/abs/2606.29959, arXiv:2606.29959

[6] Izacard G, Grave E (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. Association for Computational Linguistics, pp 874–880, https://doi.org/10.18653/v1/2021.eacl-main.74

[7] Izacard G, Caron M, Hosseini L, et al (2022) Unsupervised Dense Information Retrieval with Contrastive Learning. Transactions on Machine Learning Research URL https://openreview.net/forum?id=jKN1pXi7b0

[8] Jeong S, Baek J, Cho S, et al (2024) Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). Association for Computational Linguistics, pp 7036–7050, https://doi.org/10.18653/v1/2024.naacl-long.389

[9] Jiang Z, Xu F, Gao L, et al (2023) Active Retrieval Augmented Generation. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp 7969–7992, https://doi.org/10.18653/v1/2023.emnlp-main.495

[10] Joshi M, Choi E, Weld D, et al (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). Association for Computational Linguistics, pp 1601–1611, https://doi.org/10.18653/v1/P17-1147

[11] Kamthan A (2026) AB-RAG: Adaptive Budgeted Retrieval-Augmented Generation for Reliable Question Answering. arXiv preprint arXiv:2606.29090, URL https://arxiv.org/abs/2606.29090, arXiv:2606.29090

[12] Karpukhin V, Oguz B, Min S, et al (2020) Dense Passage Retrieval for Open-Domain Question Answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, pp 6769–6781, https://doi.org/10.18653/v1/2020.emnlp-main.550

[13] Kwon W, Li Z, Zhuang S, et al (2023) Efficient Memory Management for Large Language Model Serving with PagedAttention. In: Proceedings of the 29th Symposium on Operating Systems Principles. ACM, pp 611–626, https://doi.org/10.1145/3600006.3613165

[14] Lewis P, Perez E, Piktus A, et al (2020) Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In: Advances in Neural Information Processing Systems, pp 9459–9474, URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

[15] Li B, Tian T, Xu Z, et al (2026) Modeling Uncertainty Trends for Timely Retrieval in Dynamic RAG. Proceedings of the AAAI Conference on Artificial Intelligence 40(37):31527–31535. https://doi.org/10.1609/aaai.v40i37.40418

[16] Moskvoretskii V, Marina M, Salnikov M, et al (2025) Adaptive Retrieval Without Self-Knowledge? Bringing Uncertainty Back Home. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, pp 6355–6384, https://doi.org/10.18653/v1/2025.acl-long.319

[17] Ni S, Bi K, Guo J, et al (2024) When Do LLMs Need Retrieval Augmentation? Mitigating LLMs' Overconfidence Helps Retrieval Augmentation. In: Findings of the Association for Computational Linguistics: ACL 2024. Association for Computational Linguistics, pp 11375–11388, https://doi.org/10.18653/v1/2024.findings-acl.675

[18] Qian P, Wang S, Peng C, et al (2026) When Should Active RAG Retrieve? A Budget-Aware Evaluation of Utility, Calibration, and Cost. arXiv preprint arXiv:2607.24010, URL https://arxiv.org/abs/2607.24010, arXiv:2607.24010

[19] Qwen Team (2024) Qwen2.5: A Party of Foundation Models. URL https://qwenlm.github.io/blog/qwen2.5/

[20] Soudani H, Kanoulas E, Hasibi F (2025) Why Uncertainty Estimation Methods Fall Short in RAG: An Axiomatic Analysis. In: Findings of the Association for Computational Linguistics: ACL 2025. Association for Computational Linguistics, pp 16596–16616, https://doi.org/10.18653/v1/2025.findings-acl.852

[21] Su W, Tang Y, Ai Q, et al (2024) DRAGIN: Dynamic Retrieval Augmented Generation based on the Real-time Information Needs of Large Language Models. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, pp 12991–13013, https://doi.org/10.18653/v1/2024.acl-long.702

[22] Trivedi H, Balasubramanian N, Khot T, et al (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). Association for Computational Linguistics, pp 10014–10037, https://doi.org/10.18653/v1/2023.acl-long.557

[23] Wang Y, Wei L, Ling H (2026) Retrieval as a Decision: Training-Free Adaptive Gating for Efficient RAG. Transactions on Machine Learning Research URL https://openreview.net/forum?id=L8gYtUZfVU

[24] Yang Z, Qi P, Zhang S, et al (2018) HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp 2369–2380, https://doi.org/10.18653/v1/D18-1259

[25] Yao Z, Qi W, Pan L, et al (2025) SeaKR: Self-aware Knowledge Retrieval for Adaptive Retrieval Augmented Generation. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, pp 27022–27043, https://doi.org/10.18653/v1/2025.acl-long.1312

[26] Zhang Z, Fang M, Chen L (2024) RetrievalQA: Assessing Adaptive Retrieval-Augmented Generation for Short-form Open-Domain Question Answering. In: Findings of the Association for Computational Linguistics: ACL 2024. Association for Computational Linguistics, pp 6963–6975, https://doi.org/10.18653/v1/2024.findings-acl.415

[27] Zheng L, Yin L, Xie Z, et al (2024) SGLang: Efficient Execution of Structured Language Model Programs. In: Advances in Neural Information Processing Systems, pp 62557–62583, https://doi.org/10.52202/079017-2000

[28] Zubkova H, Park JH, Lee SW (2025) SUGAR: Leveraging Contextual Confidence for Smarter Retrieval. In: ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp 1–5, https://doi.org/10.1109/icassp49660.2025.10890064

[29] Zubkova H, Park JH, Lee SW (2026) Leveraging contextual confidence for smarter retrieval in large language models. Neural Networks 200:108862. https://doi.org/10.1016/j.neunet.2026.108862