Gradient Boosted Trees for Retrieval-Augmented Generation

Huaiyu Qin Chunyu Wei[†] **Yueguo Chen Yunhai Wang**
School of Information, Renmin University of China, Beijing, China
{qin.huaiyu, weichunyu, chenyueguo, wang.yh}@ruc.edu.cn

Abstract

Retrieval-augmented generation (RAG) divides into two camps failing in complementary ways on broad, long-tailed questions: *iterative* agentic RAG re-queries a biased distribution and drifts on prominent aspects, while *structural* RAG commits to a hierarchy fixed upfront and cannot admit aspects it initially missed. We propose **GBT-RAG**, which unifies the two by transposing gradient boosting into the text domain: each round is a query decomposition tree fitted to its predecessors' residual gap. This raises two challenges mirroring the pillars of boosting—natural-language critiques are not quantifiable, and successive rounds drift rather than fit the residual. We address the first with a *text gradient*, a structured residual in the retriever's embedding space whose support localizes missingness to specific leaves; we address the second by growing each tree only over under-covered leaves and admitting each rewrite through a monotone-improvement gate. Across standard benchmarks, GBT-RAG consistently outperforms the strongest iterative and structural baselines, with the largest gains on the long-tailed regime that motivated the design. [1]

1 Introduction

Retrieval-augmented generation (RAG) grounds large language models in external knowledge, supplying the factual accuracy, freshness, and verifiability that parametric memory cannot. As queries shift from single-hop lookup to open-ended questions spanning many documents and reasoning steps, two paradigms have emerged. *Iterative* agentic RAG [Yu et al., 2024, Asl et al., 2025] interleaves retrieval, generation, and self-critique across rounds, letting the agent decide when its answer is insufficient. *Structural* RAG [Jiao et al., 2026, Hu et al., 2025] imposes hierarchy over the corpus or the query, on the premise that complex questions require decomposition rather than flat retrieval.

Yet these two paradigms have never been properly unified. Iterative methods treat each round as a monolithic re-query, with no internal structure to direct where the next round should look; structural methods fix their hierarchy once and for all, with no mechanism to refine it as evidence accumulates. Consider a broad question such as *"What are the long-term risks of taking aspirin?"*, whose faithful answer must cover gastrointestinal effects, major bleeding, allergic reactions such as AERD, and rare but critical complications such as Reye's syndrome. The corpus is heavily long-tailed: prominent aspects dominate retrieval while rare ones sit in a sparse tail that no paraphrase of the original query can reach. Iterative agents re-sample the same biased distribution every round, deepening coverage of what is already covered; structural agents bake the bias into a hierarchy fixed at $t{=}0$, so a missed aspect can never become a leaf (Figure 1). A capable RAG agent requires both: structure to localize what is missing, and iteration to act on it. The absence of this union is the gap our work addresses.

We propose **Gradient Boosting Trees for RAG (GBT-RAG)**, drawing inspiration from the classical gradient-boosting tree, which composes weak learners into a strong predictor by fitting each new

[†]Chunyu Wei is the corresponding author.
[1]Code is available at `https://github.com/qhyyyyy/GBT-RAG`.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

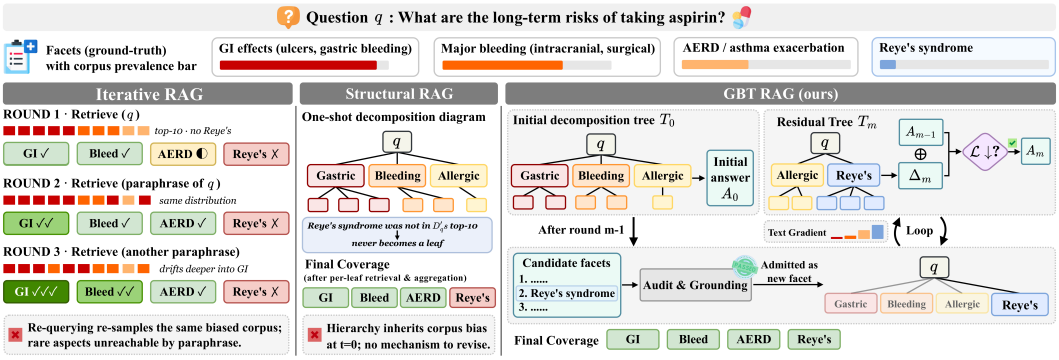

Figure 1: **Three RAG paradigms on a broad, long-tailed question.** Iterative RAG drifts on prominent aspects; structural RAG freezes a biased hierarchy; only GBT-RAG localizes the residual and admits the missing aspect.

learner to the residual errors of its predecessors. In GBT-RAG, each round of retrieval is a query decomposition tree that serves as a weak learner; successive trees are fitted to the residual gap of previous ones; and the ensemble progressively targets missing aspects. This makes iteration and structure mutually reinforcing: structure determines *where* the residual lives, and iteration targets the remaining gap.

Realizing this vision confronts two challenges that mirror the two pillars of gradient boosting. **The gradient challenge**: feedback is not quantifiable. Natural-language self-critiques do not by themselves provide persistent numerical leaf coordinates, so a structured residual is needed. **The boosting challenge**: updates are not disciplined. Successive rounds tend to paraphrase the original query rather than fit the residual, and accept every update without verifying improvement, so iteration drifts or silently regresses.

To address the **gradient challenge**, we instantiate the weak learner as a query decomposition tree whose leaves carry explicit descriptions of what each sub-aspect should cover. Comparing leaf descriptions against the current answer in the retriever's embedding space yields a **text gradient**: a structured residual whose coordinates localize missingness to specific leaves and whose magnitudes are commensurable across rounds, turning vague critique into a signal one can actually descend.

To address the **boosting challenge**, we grow each subsequent tree only over the under-covered leaves of its predecessors, so every round is fitted to the residual rather than re-litigating what is already answered. Each update is admitted only when the current-round aggregate coverage loss does not increase and rolled back otherwise, yielding non-increase on the measured coverage signal.

Our contributions are threefold.

- We identify the lack of integration between iterative and structural RAG as the key bottleneck in current agentic systems.
- We propose GBT-RAG, which realizes this integration through the three ingredients of gradient boosting in the text domain: trees as weak learners, a text gradient as a structured residual over their leaves, and a descent-gated boosting loop.
- We demonstrate consistent gains over strong iterative agentic RAG baselines on multi-hop and open-domain benchmarks, with interpretable per-aspect traces of how each tree contributes to closing the gap.

2 Related Work

Iterative and agentic retrieval-augmented generation. Moving beyond single-pass retrieve-then-read [Lewis et al., 2020, Guu et al., 2020, Izacard and Grave, 2021], iterative methods interleave retrieval and generation. Active retrieval uses uncertainty or confidence to re-query [Jiang et al., 2023, Su et al., 2024, Jeong et al., 2024], while agentic frameworks let LMs control retrieval via reasoning [Yao et al., 2023b, Schick et al., 2023, Trivedi et al., 2023, Press et al., 2023, Khattab

et al., 2022]. Self-corrective variants add verification and rewriting [Asai et al., 2024, Yan et al., 2024, Madaan et al., 2023, Shinn et al., 2023, Gou et al., 2024, Qin et al., 2026], and recent systems chain multi-round search with explicit termination [Shao et al., 2023, Yu et al., 2024, Asl et al., 2025, Li et al., 2025, Wang et al., 2024]. However, treating each round as a monolithic re-query with non-quantifiable free-form critiques motivates our proposed text gradient.

Structural decomposition for retrieval-augmented generation. A complementary line adds corpus or query hierarchy. Corpus-side methods use multi-resolution summaries [Sarthi et al., 2024] or entity graphs [Edge et al., 2024, Guo et al., 2025, Gutierrez et al., 2024, Peng et al., 2026, Han et al., 2025, Wei et al., 2026]. Query-side methods split complex questions into independent sub-questions [Khot et al., 2023, Zhou et al., 2023, Min et al., 2019, Perez et al., 2020, Radhakrishnan et al., 2023], often using tree search [Yao et al., 2023a, Hu et al., 2025, Besta et al., 2024] or pruning [Jiao et al., 2026]. Related tactics include step-back prompting [Zheng et al., 2024], HyDE [Gao et al., 2023], and CoVe [Dhuliawala et al., 2024]. Because these fix hierarchy upfront and inherit initial retrieval bias, GBT-RAG instead adaptively grows decompositions using a residual signal from the current answer.

Gradient boosting and text-domain optimization. We build on classical gradient boosting [Friedman, 2001, Mason et al., 1999], dominating structured prediction [Chen and Guestrin, 2016, Ke et al., 2017, Prokhorenkova et al., 2018] since Freund and Schapire [1997] and Schapire [1990]. Recently, text-domain gradient updates have emerged: ProTeGi optimizes prompts via LM critiques [Pryzant et al., 2023], TextGrad propagates language feedback [Yüksekgönül et al., 2025], and DSPy optimizes modular pipelines [Khattab et al., 2024]. Other works frame self-improvement or search as functional updates [Madaan et al., 2023, Zelikman et al., 2022, Wang et al., 2023, Chen et al., 2024]. GBT-RAG uniquely instantiates boosting's core commitments residual, weak learner, and composition directly over text, yielding descent-gated retrieval-augmented generation.

3 Preliminaries

Problem Setup. In the standard RAG setting, let $\mathcal{C} = \{d_1, \ldots, d_N\}$ be a passage corpus and $\mathcal{V}^*$ denote token sequences over vocabulary $\mathcal{V}$. For a query $q \in \mathcal{V}^*$, we seek an answer $A \in \mathcal{V}^*$ faithful to $\mathcal{C}$. The system's sole interface comprises three black-box oracles: a retriever $\mathcal{R} : \mathcal{V}^* \to 2^{\mathcal{C}}$, a language model $\mathcal{M} : \mathcal{V}^* \to \mathcal{V}^*$, and an embedding map $\phi : \mathcal{V}^* \to \mathbb{S}^{n-1} \subset \mathbb{R}^n$ (default $n = 768$) yielding cosine similarities $\langle \phi(x), \phi(y) \rangle$.

Gradient Boosting in Function Space. Classical gradient boosting [Friedman, 2001] builds a predictor F_M via stagewise descent on a loss $\mathcal{L}$, composing weak learners from $\mathcal{H}$:

$$F_m \;=\; F_{m-1} + \nu_m \cdot h_m, \qquad h_m \;\in\; \arg\min_{h \in \mathcal{H}} \big\| -\nabla \mathcal{L}(F_{m-1}) - h \big\|^2, \tag{1}$$

where h_m regresses the negative functional gradient with step size $\nu_m > 0$. Eq. (1) requires three commitments: **(C1)** a differentiable loss $\mathcal{L}$ inducing an improvement-oriented residual; **(C2)** a constrained weak-learner class $\mathcal{H}$ regressing this residual; and **(C3)** a composition operator $\oplus$ where $F_m = F_{m-1} \oplus (\nu_m h_m)$. Because RAG's text space $\mathcal{V}^*$ lacks vector-space structure and differentiability, our contribution instantiates (C1)–(C3) over $\mathcal{V}^*$ while preserving monotone descent.

Decomposition Trees. Our weak learners are query decomposition trees, serving simultaneously as a retrieval planner and a coordinate system for measuring coverage. A *decomposition tree* is a tuple $T = (V, E, q, \rho, \sigma)$, where (V, E) is a rooted tree, $q \in V$ is the root, $\rho : V \to \mathcal{V}^*$ assigns each node a sub-question, and $\sigma : \mathcal{L}(T) \to \mathcal{V}^*$ assigns each leaf ℓ a *sub-aspect description* $s_\ell = \sigma(\ell)$. Each leaf further induces a retrieval query q_ℓ and an evidence string e_ℓ, both derived from s_ℓ during execution. We restrict attention to depth-two trees, with D children at the root (*dimensions*) and a finite sub-aspect set S_d under each dimension d, giving $K = \sum_{d=1}^{D} |S_d|$ leaves. Critique-driven decomposition determines these sizes from the query; a uniform $D \times S$ decomposition is a special case. Decomposition trees thus serve as the weak-learner class $\mathcal{H}$ of Eq. (1); how successive trees compose is the subject of Section 4.

4 Methodology

We instantiate (C1)–(C3) over text answers, yielding the GBT-RAG framework in Figure 2. (C1) is discharged by a coverage residual derived from the embedding geometry of the leaf set, supplying

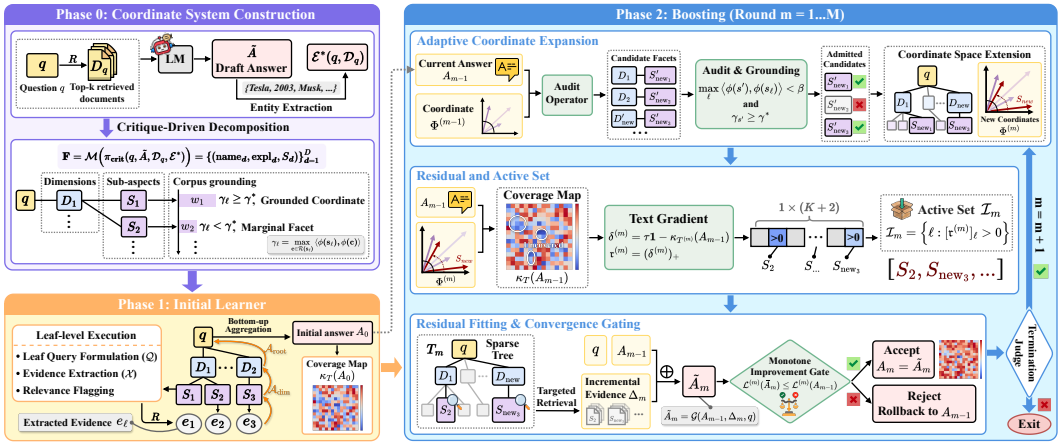

Figure 2: Overview of GBT-RAG. Phase 0 builds the coordinate system $\Phi(T_0)$; Phase 1 executes the initial tree T_0 to produce A_0; each round of Phase 2 fits a residual tree T_m over the active set $\mathcal{I}_m$ and admits the rewrite $\tilde{A}_m$ via a monotone-improvement gate.

a structured signal replacing the missing functional gradient (Section 4.1). (C2) is discharged by decomposition trees grown adaptively over the active leaves of this residual (Sections 4.2–4.4). (C3) is discharged by a monotone-gated rewrite operator that precludes increases in current-round aggregate coverage loss (Section 4.4).

4.1 The Coverage Residual and Loss

We first convert a decomposition tree into a finite coordinate system against which any candidate answer can be measured. The leaf set enumerates aspects a faithful answer should cover, and ϕ represents each as a unit vector. Stacking these columnwise yields

$$\Phi(T) \;=\; \big[\,\phi(s_1),\,\phi(s_2),\,\ldots,\,\phi(s_K)\,\big] \;\in\; \mathbb{R}^{n\times K}, \tag{2}$$

which we treat as a dictionary of leaf directions: any candidate is characterized by its similarities to these columns.

To make this projection precise, assume any candidate $A \in \mathcal{V}^*$ admits a **sentence segmentation** $A = \bigsqcup_{j=1}^{P} p_j$, with associated embedding cloud $\{\phi(p_j)\}_{j=1}^{P} \subset \mathbb{S}^{n-1}$. The *coverage map* $\kappa_T : \mathcal{V}^* \to [-1,1]^K$ is defined coordinate-wise by

$$[\kappa_T(A)]_\ell \;=\; \max_{j\in[P]} \big\langle \phi(p_j),\,\phi(s_\ell)\big\rangle, \qquad \ell \in [K], \tag{3}$$

where $[P] = \{1,\ldots,P\}$. The maximum captures that a sub-aspect is covered as soon as one aligns sufficiently with it; equivalently, $[\kappa_T(A)]_\ell$ is the ℓ^∞-projection of the embedded cloud onto $\phi(s_\ell)$. Unit normalization bounds these similarities in $[-1,1]$; it does not ensure stability under textual edits.

We turn the coverage map into a scalar loss. Let $\tau \in (0,1)$ be a target margin and $\mathbf{w} \in \mathbb{R}^K_{>0}$ a leaf-importance weight vector. The *coverage residual* $\mathfrak{r}_T$ and *coverage loss* $\mathcal{L}_T$ are

$$\mathfrak{r}_T(A) \;=\; \big(\tau\mathbf{1} - \kappa_T(A)\big)_+, \qquad \mathcal{L}_T(A) \;=\; \big\langle \mathbf{w}, \mathfrak{r}_T(A)\big\rangle \;=\; \sum_{\ell=1}^{K} w_\ell \cdot \big(\tau - [\kappa_T(A)]_\ell\big)_+, \tag{4}$$

where $(\cdot)_+ = \max(\cdot, 0)$. The loss is a weighted hinge functional: it saturates to zero on coordinates exceeding τ, suppressing wasted retrieval, and grows linearly below the margin, preserving an actionable signal where the answer is deficient. Crucially, $\mathfrak{r}_T(A) \in \mathbb{R}^K_{\geq 0}$ plays the role of $-\nabla\mathcal{L}(F)$ in Eq. (1): its support identifies leaves the next learner should target, and its magnitudes are commensurable across leaves and rounds because they share one embedding geometry. We refer to $\mathfrak{r}_T(A)$ as the *text gradient*; it is an operational residual rather than a literal hinge-loss derivative (Appendix H).

4.2 Phase 0: Constructing the Coordinate System

The text gradient is meaningful only if $\Phi(T_0)$ faithfully reflects query-relevant aspects, so Phase 0 constructs this coordinate system together with the calibration weights $\mathbf{w}$, via critique-grounded decomposition that combines a draft answer with corpus evidence.

We obtain a top-k retrieval $\mathcal{D}_q = \mathcal{R}(q)$ with $k = 10$ and elicit a draft $\tilde{A} = \mathcal{M}\big(\pi_{\mathrm{draft}}(q, \mathcal{D}_q)\big)$, where π_{draft} is a fixed prompt template. To stabilize subsequent generations against entity drift, we extract candidate entities via a regex filter $\mathcal{E}$, verify them against $\mathcal{D}_q$, and propagate the verified set $\mathcal{E}^*(q, \mathcal{D}_q)$ to all downstream prompts. The model then enumerates facets and their sub-aspects:

$$\mathbf{F} \;=\; \mathcal{M}\big(\pi_{\mathrm{crit}}(q, \tilde{A}, \mathcal{D}_q, \mathcal{E}^*)\big) \;=\; \big\{(\mathrm{name}_d,\, \mathrm{expl}_d,\, S_d)\big\}_{d=1}^{D}, \tag{5}$$

where D is the number of returned dimensions, $\mathrm{name}_d, \mathrm{expl}_d$ are its name and explanation, and S_d contains its returned sub-aspects. Flattening yields the leaf set $\{s_\ell\}_{\ell=1}^{K_0}$ with $K_0 = \sum_{d=1}^{D} |S_d|$, identified columnwise with $\Phi(T_0)$ in Eq. (2).

The leaf set reflects what $\mathcal{M}$ believes a complete answer should cover, but not all such facets are demonstrably retrievable; equal weighting would let unsupported aspects dominate the loss. We instantiate $\mathbf{w}$ via a corpus-grounding test:

$$\gamma_\ell \;=\; \max_{c \in \mathcal{R}(s_\ell)} \big\langle \phi(s_\ell),\, \phi(c) \big\rangle, \qquad w_\ell \;=\; \begin{cases} 1.0 & \text{if } \gamma_\ell \;\geq\; \gamma^*, \\ 0.5 & \text{if } \gamma_\ell \;<\; \gamma^*, \end{cases} \tag{6}$$

where γ^* is a fixed grounding threshold. Leaves demonstrably retrievable contribute fully; marginal facets contribute at half weight, preventing weakly grounded dimensions from dominating $\mathcal{L}_T$ while still allowing them to be tested in later rounds.

4.3 Phase 1: The Initial Learner T_0

Given $\Phi(T_0)$ from Phase 0, we instantiate the initial weak learner T_0 as the depth-two tree of Section 3 whose dimensions correspond to $\mathbf{F}$ and whose leaves are $\{s_\ell\}$. Execution proceeds bottom-up by composing three operators per leaf:

$$\mathrm{q}_\ell \;=\; \mathcal{Q}(s_\ell), \qquad E_\ell \;=\; \mathrm{Top}_k\big(\mathcal{R}(\mathrm{q}_\ell)\big), \qquad e_\ell \;=\; \mathcal{X}(s_\ell, E_\ell), \tag{7}$$

where $\mathcal{Q}$ distills s_ℓ into a factoid retrieval query, Top_k selects the top-k passages (optionally with cross-encoder reranking), and $\mathcal{X}$ extracts a verbatim evidence string e_ℓ. Leaf evidence is aggregated upward to dimensions and then to the root:

$$u_d \;=\; \mathcal{A}_{\mathrm{dim}}\big(\mathrm{name}_d, \{e_\ell\}_{\ell \in S_d}\big), \quad d \in [D], \qquad A_0 \;=\; \mathcal{A}_{\mathrm{root}}\big(q, \{u_d\}_{d=1}^{D}\big), \tag{8}$$

where both aggregators are entity-preserving: paraphrasing of named entities, dates, and numerical quantities is forbidden. The triple $(\Phi(T_0), \mathbf{w}, A_0)$ initializes the boosting loop.

4.4 Phase 2: The Boosting Loop

The core of GBT-RAG is a boosting loop that, at each round, computes the text gradient on the current answer, fits a residual tree to the active leaves, and rewrites the answer subject to a monotone-improvement gate. We describe round m, with A_{m-1}, $\Phi(T_{m-1})$, and $\mathbf{w}^{(m-1)}$ given.

Adaptive coordinate expansion. The hypothesis space of facets is adaptive: as the answer evolves, aspects may emerge. We equip the loop with an audit operator proposing candidate facets,

$$\mathbf{F}_m^{\mathrm{new}} \;=\; \mathcal{M}\big(\pi_{\mathrm{audit}}(A_{m-1}, \{s_\ell\})\big), \tag{9}$$

where π_{audit} exposes $\mathcal{M}$ to the current answer and existing leaves. Each candidate s' must be sufficiently distinct and demonstrably grounded:

$$\max_{\ell} \big\langle \phi(s'),\, \phi(s_\ell) \big\rangle \;<\; \beta \qquad \text{and} \qquad \gamma_{s'} \;\geq\; \gamma^*, \tag{10}$$

where $\beta \in (0, 1)$ is a deduplication threshold and $\gamma_{s'}$ is computed as in Eq. (6). Passing candidates are appended to yield $\Phi^{(m)}$ with corresponding weights, with a hard cap $K_{\max}$ on total leaves preventing unbounded growth.

Residual and active set. With $\Phi^{(m)}$ in place, we evaluate the text gradient:

$$\mathfrak{r}^{(m)} \;=\; \big(\tau\mathbf{1} - \kappa_{T^{(m)}}(A_{m-1})\big)_+ \;\in\; \mathbb{R}^{K_m}_{\geq 0}, \qquad \mathcal{I}_m \;=\; \big\{\ell : [\mathfrak{r}^{(m)}]_\ell > 0\big\}, \tag{11}$$

where $\mathcal{I}_m \subseteq [K_m]$ is the *active set*: the discrete analogue of the support of $-\nabla\mathcal{L}(F_{m-1})$ in Eq. (1). To prevent capture by leaves the corpus cannot support, we apply *dead-leaf pruning*: ℓ is removed from $\Phi^{(m)}$ if both $[\kappa(A_{m-1})]_\ell < \kappa_{\min}$ and $\max_{m'<m,\, \ell' \in T_{m'}} \langle \phi(e_{\ell'}), \phi(s_\ell)\rangle < \kappa_{\text{evid}}$. Persistent non-coverage with retrieval failure motivates this pruning heuristic; removing a leaf does not establish coverage.

Residual tree as a weak learner. With $\mathcal{I}_m$ identified, we build a depth-two residual tree:

$$T_m = \text{BuildResidualTree}\big(\{(\ell, s_\ell) : \ell \in \mathcal{I}_m\}\big), \qquad \text{LeafIDs}(T_m) = \mathcal{I}_m. \tag{12}$$

Active leaves retain their identifiers and are grouped under parent dimensions. The restricted class $\mathcal{H}_{\mathcal{I}_m}$ contains residual trees with nonempty leaves from $\mathcal{I}_m$; it is empty when $\mathcal{I}_m$ is empty. This is evidence collection on residual support, not numerical functional-gradient regression. Executing T_m via Eq. (7) produces the round-m evidence increment

$$\Delta_m \;=\; \big\{e_\ell : \ell \in \mathcal{I}_m\big\}, \tag{13}$$

the textual analogue of $\nu_m h_m$ in Eq. (1): a fitted weak learner targeting only the residual.

Functional composition with monotone gating. Composition is realized by an evidence-grounded rewrite:

$$\tilde{A}_m \;=\; \mathcal{G}\big(A_{m-1},\, \Delta_m,\, q\big), \tag{14}$$

where $\mathcal{G}$ is a length-capped, entity-preserving generation operator that integrates Δ_m into A_{m-1} while remaining anchored on q. Since $\mathcal{G}$ is black-box, $\tilde{A}_m$ may increase the loss; an ungated rule would inherit the silent-regression failure mode that motivated our framework. We thus admit $\tilde{A}_m$ only when the current-round aggregate coverage loss does not increase:

$$A_m \;=\; \begin{cases} \tilde{A}_m, & \text{if } \mathcal{L}^{(m)}(\tilde{A}_m) \;\leq\; \mathcal{L}^{(m)}(A_{m-1}), \\ A_{m-1}, & \text{otherwise (rollback)}, \end{cases} \tag{15}$$

where $\mathcal{L}^{(m)}$ is the coverage loss of Eq. (4) on $\Phi^{(m)}$ with weights $\mathbf{w}^{(m)}$. The conditional in Eq. (15) constitutes the operator $\oplus$ of (C3) and ensures non-increase when both answers are evaluated on the same round-m coordinates.

Termination. The loop terminates at round m when one of four conditions holds: *absolute convergence* ($\mathcal{L}^{(m)}(A_m) < \varepsilon$); *stagnation after acceptance* ($\mathcal{L}^{(m)}(A_{m-1}) - \mathcal{L}^{(m)}(A_m) < \delta$); *empty active set* ($\mathcal{I}_m = \emptyset$); or *budget exhaustion* ($m = M$). A rollback round does not by itself terminate the loop: the stagnation check is bypassed and the next round computes a fresh residual on the unchanged answer, possibly with an expanded coordinate system. The cap M combined with Eq. (15) guarantees finite-time termination. The full procedure is summarized in Appendix B.

5 Theoretical Analysis

We establish two properties of the gate in Eq. (15): current-round aggregate non-increase, and inherited-leaf non-increase under assumptions (A1) and (A2) below. The gate alone does not ensure per-leaf preservation across adaptive expansion.

Theorem 5.1 (Monotone descent of GBT-RAG)**.** *Let $\Phi^{(m)}$ with weights $\mathbf{w}^{(m)} \in \mathbb{R}^{K_m}_{>0}$ be the round-m coordinate system, $\kappa^{(m)} = \kappa_{T^{(m)}}$ the coverage map of Eq.* (3)*, and*

$$\mathcal{L}^{(m)}(A) \;=\; \sum_{\ell \in [K_m]} w^{(m)}_\ell \big(\tau - [\kappa^{(m)}(A)]_\ell\big)_+, \qquad \mathcal{L}^{(m)}_{\mathcal{S}}(A) \;=\; \sum_{\ell \in \mathcal{S}} w^{(m)}_\ell \big(\tau - [\kappa^{(m)}(A)]_\ell\big)_+ \tag{16}$$

the round-m loss and its restriction to $\mathcal{S} \subseteq [K_m]$. Assume (A1) *expansion appends leaves: for every $m \geq 1$ there is an inclusion $[K_{m-1}] \hookrightarrow [K_m]$ under which $\phi(s_\ell)$ and $w^{(m)}_\ell$ coincide with their round-$(m-1)$ counterparts for $\ell \in [K_{m-1}]$; and* (A2) *$\mathcal{G}$ is evidence-monotone: $\mathcal{P}(A_{m-1}) \subseteq \mathcal{P}(\tilde{A}_m)$ when $\tilde{A}_m = \mathcal{G}(A_{m-1}, \Delta_m, q)$, where $\mathcal{P}(A) = \{\phi(p_j)\}_{j=1}^P$. Then for every $m \geq 1$,*

$$\mathcal{L}^{(m)}(A_m) \;\leq\; \mathcal{L}^{(m)}(A_{m-1}), \tag{17}$$

$$\mathcal{L}^{(m)}_{\mathcal{S}}(A_m) \;\leq\; \mathcal{L}^{(m-1)}_{\mathcal{S}}(A_{m-1}) \qquad \forall\, \mathcal{S} \subseteq [K_{m-1}], \tag{18}$$

with equality in (17) *on rollback rounds. In particular, the loss contributed by $\Phi^{(0)}$ is non-increasing along the entire trajectory.*

Proof. Per-round (17). The gate of Eq. (15) sets $A_m = \tilde{A}_m$ when $\mathcal{L}^{(m)}(\tilde{A}_m) \le \mathcal{L}^{(m)}(A_{m-1})$, and $A_m = A_{m-1}$ otherwise. On accepted rounds $\mathcal{L}^{(m)}(A_m) \le \mathcal{L}^{(m)}(A_{m-1})$; on rollback rounds equality holds.

Cross-round (18). Fix $\mathcal{S} \subseteq [K_{m-1}]$. By (A1), $\phi(s_\ell)$ is invariant in m on $\mathcal{S}$, so

$$[\kappa^{(m)}(A)]_\ell \;=\; \max_{j\in[P]} \big\langle \phi(p_j), \phi(s_\ell) \big\rangle \;=\; [\kappa^{(m-1)}(A)]_\ell, \qquad \ell \in \mathcal{S}, \tag{19}$$

hence $\mathcal{L}_{\mathcal{S}}^{(m)}(A) = \mathcal{L}_{\mathcal{S}}^{(m-1)}(A)$ for every A. By (A2), $\mathcal{P}(A_{m-1}) \subseteq \mathcal{P}(\tilde{A}_m)$, so

$$[\kappa^{(m)}(\tilde{A}_m)]_\ell \;=\; \max_{\phi(p)\in\mathcal{P}(\tilde{A}_m)} \langle \phi(p), \phi(s_\ell)\rangle \;\ge\; \max_{\phi(p)\in\mathcal{P}(A_{m-1})} \langle \phi(p), \phi(s_\ell)\rangle \;=\; [\kappa^{(m)}(A_{m-1})]_\ell \tag{20}$$

for $\ell \in [K_m]$. Since $x \mapsto (\tau - x)_+$ is non-increasing, summing over $\mathcal{S}$ with $w_\ell^{(m)} = w_\ell^{(m-1)}$ yields

$$\mathcal{L}_{\mathcal{S}}^{(m)}(\tilde{A}_m) \;\le\; \mathcal{L}_{\mathcal{S}}^{(m)}(A_{m-1}) \overset{(19)}{=} \mathcal{L}_{\mathcal{S}}^{(m-1)}(A_{m-1}). \tag{21}$$

On accepted rounds $A_m = \tilde{A}_m$, giving (18); on rollback rounds the bound reduces to (19). The final claim follows by taking $\mathcal{S} = [K_0]$ and chaining across $m = 1, \dots, M$. □

Theorem 5.1 formalizes the gating discipline. The per-round claim follows from the gate alone; the inherited-leaf claim additionally requires (A1) and (A2). A scalar gate can accept a loss on one leaf if others improve sufficiently. Pruned leaves fall outside append-only persistence, and newly added coordinates can increase total loss. Appendix H discusses these limits; Appendix A gives finite termination and the zero-residual characterization on retained coordinates.

6 Experiments

6.1 Experimental Setup

We evaluate on three benchmarks: **HotpotQA** [Yang et al., 2018] for two-hop reasoning, **MuSiQue** [Trivedi et al., 2022] for shortcut-resistant 2–4-hop questions, and **ASQA** [Stelmakh et al., 2022] for ambiguous questions requiring multiple disambiguated interpretations. We report token-level F1/EM on HotpotQA and MuSiQue, and Disambig-F1, ROUGE-L, and DR score on ASQA (details in Appendices D–F). Baselines include **Standard RAG** [Lewis et al., 2020]; iterative methods **FLARE** [Jiang et al., 2023], **Iter-Retgen** [Shao et al., 2023], **Auto-RAG** [Yu et al., 2024], **FAIR-RAG** [Asl et al., 2025]; and structural methods **MCTS-RAG** [Hu et al., 2025], **PruneRAG** [Jiao et al., 2026] (Appendix E). All methods run on Llama-3.1-8B-Instruct and Qwen2.5-7B-Instruct with retrieval encoder `e5-base-v2` [Wang et al., 2022] over the Dec-2018 DPR Wikipedia corpus [Karpukhin et al., 2020] (embeddings for Eq. (3)), using FlashRAG [Jin et al., 2025] to ensure reproducible comparisons. GBT-RAG hyperparameters are fixed across datasets (Appendix C). Table 1 reports means over five evaluation-question samples; ASQA uses 500 draws with replacement per sample from its 948-question development pool (Appendix D).

6.2 Main Results

Table 1 reports the main comparison; we highlight three observations.

(1) GBT-RAG dominates on coverage-sensitive metrics. On HotpotQA F1/EM, MuSiQue F1/EM, and ASQA Disambig-F1 metrics rewarding *all* required hops or facets GBT-RAG beats every baseline across both backbones. It tops PruneRAG by $+7.4$ F1 on HotpotQA and $+19.6$ F1 on MuSiQue (Llama-3.1-8B), and $+12.4$ and $+12.2$ F1 (Qwen2.5-7B). The MuSiQue margin nearly doubling the top structural baseline shows that on highly compositional questions, paraphrase-based methods miss long-tailed hops the residual directly finds.

(2) Iterative and structural baselines fail in complementary ways, as Figure 1 predicts. On Llama-3.1-8B, iterative methods span 31.4–50.0 F1 on HotpotQA and 15.1–28.4 F1 on MuSiQue:

Table 1: Mean results over five evaluation-question samples on HotpotQA, ASQA, and MuSiQue under two backbones. **Bold** marks the best result per column within a backbone; underline marks the second-best.

Backbone	Type	Method	HotpotQA		ASQA			MuSiQue	
			F1	EM	DR	ROUGE-L	Disambig	F1	EM
Llama-3.1-8B	Sequential	Standard RAG	37.5	26.1	30.1	26.5	34.2	13.5	8.4
	Iterative	FLARE	31.4	25.7	31.4	**29.7**	33.1	15.1	9.6
		Iter-Retgen	50.0	36.1	29.3	24.2	35.4	24.2	16.9
		Auto-RAG	48.3	40.6	29.1	25.9	32.6	28.4	19.7
		FAIR-RAG	46.2	37.3	30.3	25.2	36.4	27.7	17.8
	Structural	MCTS-RAG	56.2	47.6	31.5	29.0	34.1	26.3	19.7
		PruneRAG	62.4	54.1	33.9	27.8	41.4	30.3	25.1
	Ours	**GBT-RAG**	**69.8**	**54.3**	**38.2**	27.9	**52.2**	**49.9**	**41.3**
Qwen2.5-7B	Sequential	Standard RAG	43.7	30.8	24.8	24.7	24.9	10.4	8.1
	Iterative	FLARE	32.6	23.3	24.8	23.4	26.3	11.5	7.2
		Iter-Retgen	47.0	34.6	30.8	**30.2**	31.4	21.8	14.7
		Auto-RAG	48.2	36.9	29.1	25.5	33.2	20.2	13.4
		FAIR-RAG	48.5	36.1	31.0	26.1	36.9	20.3	14.3
	Structural	MCTS-RAG	50.1	42.7	29.2	27.3	31.2	19.3	13.9
		PruneRAG	61.3	50.1	34.4	29.1	40.6	25.6	17.6
	Ours	**GBT-RAG**	**73.7**	**55.3**	**35.0**	26.8	**45.8**	**37.8**	**28.3**

re-querying deepens already-prominent aspects but misses rare ones. Structural baselines achieve higher scores on HotpotQA, yet PruneRAG's MuSiQue F1 of 30.3 illustrates a remaining gap for the tested one-shot decomposition. GBT-RAG closes the gap neither family closes alone.

(3) ASQA: substantive coverage at a small cost in lexical mimicry. On Disambig-F1 ASQA's core metric GBT-RAG beats PruneRAG by $+10.8$ on Llama-3.1-8B (26% relative gain) and $+5.2$ on Qwen2.5-7B, proving the residual loop finds missed interpretations. GBT-RAG scores 27.9 versus 27.8 for PruneRAG on Llama-3.1-8B, and 26.8 versus 29.1 on Qwen2.5-7B in ROUGE-L. The DR score, combining disambiguation accuracy and reference similarity, places GBT-RAG first on both backbones ($+4.3$ over PruneRAG on Llama-3.1-8B). Trading minor lexical deficits for double-digit coverage gains fits ASQA's design.

6.3 Ablation Study

We ablate five design choices targeting (C1)–(C3) and iteration. Table 2 reports ASQA results on Llama-3.1-8B-Instruct, removing one component at a time. The largest drops hit the iteration–decomposition core: *removing the residual tree* (growing over all leaves, not $\mathcal{I}_m$) costs 11.3 DR by re-litigating covered leaves; *running Phase 1 only* (without boosting updates) costs 10.7 DR, proving one-shot decomposition insufficient despite high-quality T_0. *Replacing κ_T with text-only critique* costs 9.9 DR by stripping the localizing residual. *Removing the monotone gate* costs 7.9 DR, corroborating Theorem 5.1 that unverified descent regresses on noisy rounds. *Replacing Phase 0's critique-driven decomposition* with a generic prompt costs 6.8 DR. No component dominates (each contributes 7–11 DR); the full pipeline is essential.

Table 2: ASQA ablation scores after final compression. ΔDR uses Full GBT-RAG as its reference.

Configuration	DR	R-L	Disamb.	ΔDR
Full GBT-RAG	**38.2**	**27.9**	**52.2**	—
w/o Residual Tree	26.9	26.1	27.7	−11.3
w/o Embed. Coverage	28.3	26.5	30.1	−9.9
w/o Monotone Gate	30.3	27.1	33.9	−7.9
Single Round	27.5	24.2	31.2	−10.7
w/o Draft-Critique	31.4	27.4	35.9	−6.8

6.4 Hyperparameter Sensitivity

We sweep four hyperparameters on ASQA, varying one while fixing others at defaults (Appendix C): coverage threshold τ (Eq. (4)), deduplication threshold β (Eq. (10)), round budget M, and retrieval depth k. Figure 3 reveals three trends. First, τ and β form inverted-U curves: under-thresholding yields weak residuals, while over-thresholding causes noisy retrievals (τ) or coordinate-weakening duplicates (β). Second, final DR in the budget sweep saturates monotonically; this is an observation

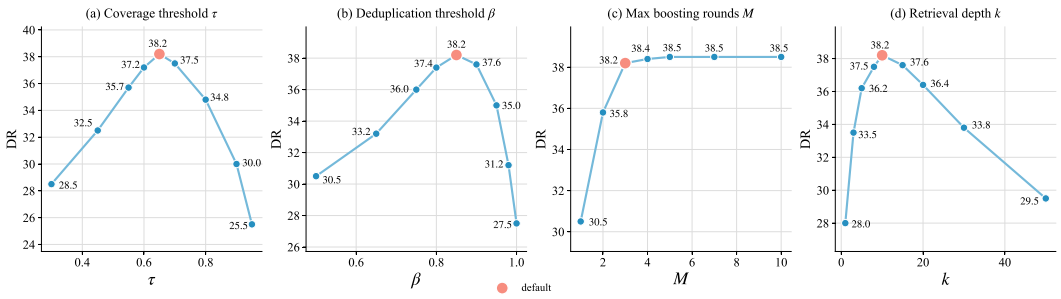

Figure 3: Hyperparameter sensitivity on ASQA (Llama-3.1-8B-Instruct). Defaults in red; panel (c) reports scores after final compression at each round budget.

about Figure 3(c), rather than a guarantee about coverage loss. Third, k shows a coverage–noise trade-off, peaking at the FlashRAG default of 10. Robustness within ± 1 DR around defaults indicates fine-grained tuning is unnecessary.

6.5 Per-Round Convergence Trace

Table 3 combines coverage diagnostics recorded during boosting with ASQA answer-quality scores evaluated after final compression. Compression is an output-stage operation; the loss and active set are computed on the uncompressed answers.

Table 3: ASQA round-wise results (Llama-3.1-8B-Instruct). DR and Disamb. are post-compression scores. Loss $\mathcal{L}^{(m)}$, active-set size $|\mathcal{I}_m|$, and cumulative retrievals describe the uncompressed boosting process.

m	$\mathcal{L}^{(m)}$	$\lvert\mathcal{I}_m\rvert$	DR	Disamb.	Cum. Ret.
0	0.45	14.8	27.5	31.2	14.8
1	0.27	7.3	30.5	35.1	22.1
2	0.14	2.4	35.8	46.5	24.5
3	0.09	0.5	38.2	52.2	25.0

under the default $M{=}3$.

The trace illustrates the observed loss and retrieval dynamics. The loss decreases monotonically from 0.45 to 0.09 on the reported round-specific coordinates; $|\mathcal{I}_m|$ shrinks from 14.8 to 0.5, contracting the support over which subsequent retrieval operates; and cumulative retrievals grow sublinearly (14.8, 22.1, 24.5, 25.0) because each round operates only on residual leaves. After compression, the first two updates achieve $+8.3$ of the total $+10.7$ DR gain, and the third adds $+2.4$, illustrating its contribution

6.6 Stratified Analysis: Where Does the Gain Come From?

To verify GBT-RAG's gains concentrate on the hardest long-tail failures (Figure 1), we stratify ASQA by interpretations and MuSiQue by hops. Figure 4 confirms this. On easy questions, GBT-RAG holds a modest lead since well-covered aspects are easily retrievable. As complexity grows, baselines collapse: from easiest to hardest strata, Standard RAG drops 17 ASQA and 19 MuSiQue points; PruneRAG drops 14 and 22. GBT-RAG stays nearly flat on ASQA ($53.4 \rightarrow 49.3$) and degrades gracefully on MuSiQue ($55.6 \rightarrow 41.2$), achieving margins of $+16.3$ Disambig-F1 on 4+-interpretation questions and $+26.5$ F1 on 4-hop MuSiQue. This widening gap confirms GBT-RAG's gains stem from the targeted long-tail regime. Appendix G provides a full ASQA trace.

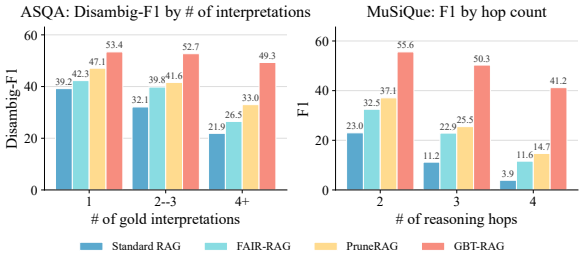

Figure 4: Stratified analysis on Llama-3.1-8B-Instruct. As complexity grows, baselines collapse while GBT-RAG holds up.

Supplementary encoder, backbone, benchmark, and budget results are reported in Appendices L–N.

7 Conclusion and Limitations

We presented GBT-RAG, a framework that unifies iterative and structural retrieval-augmented generation by transposing the three commitments of gradient boosting a residual signal, a weak learner, and a composition operator into the discrete domain of text answers: a coverage residual over an embedded leaf coordinate system as a quantifiable text gradient, decomposition trees grown adaptively over the active set as residual-fitted weak learners, and a monotone-gated rewrite operator that yields non-increasing current-round coverage loss and, under the stated assumptions, non-increasing inherited-leaf loss. Across HotpotQA, MuSiQue, and ASQA on two open-source backbones, GBT-RAG consistently outperforms the strongest iterative and structural baselines, with the largest gains on the long-tailed regime that motivated its design. Two avenues remain open: jointly adapting ϕ to the boosting trajectory (or instantiating the residual under multiple embeddings) to remove dependence on a fixed encoder geometry, and lifting the depth-two restriction to learn variable-depth structures. Improved coverage can benefit knowledge-intensive applications, while biased or false retrieved content can still be amplified; the coverage proxy does not guarantee factual correctness.

Acknowledgments and Disclosure of Funding

This research was supported by the National Key R&D Program of China (No. 2025YFC3309503), NSFC (No. 62506366), the CCF-Tencent Rhino-Bird Open Research Fund, the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China. It was also supported by Big Data and Responsible Artificial Intelligence for National Governance, Renmin University of China, and by the Public Computing Cloud, Renmin University of China.

References

A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

M. A. Asl, M. Asgari-Bidhendi, and B. Minaei-Bidgoli. FAIR-RAG: faithful adaptive iterative refinement for retrieval-augmented generation. *CoRR*, abs/2510.22344, 2025. doi: 10.48550/ARXIV.2510.22344. URL https://doi.org/10.48550/arXiv.2510.22344.

M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk, and T. Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In M. J. Wooldridge, J. G. Dy, and S. Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pages 17682–17690. AAAI Press, 2024. doi: 10.1609/AAAI.V38I16.29720. URL https://doi.org/10.1609/aaai.v38i16.29720.

T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

X. Chen, M. Lin, N. Schärli, and D. Zhou. Teaching large language models to self-debug. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=KuPixIqPiq.

S. Dhuliawala, M. Komeili, J. Xu, R. Raileanu, X. Li, A. Celikyilmaz, and J. Weston. Chain-of-verification reduces hallucination in large language models. In L. Ku, A. Martins, and V. Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Findings of ACL, pages 3563–3578. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-ACL.212. URL https://doi.org/10.18653/v1/2024.findings-acl.212.

D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitansky, R. O. Ness, and J. Larson. From local to global: A graph RAG approach to query-focused summarization. *CoRR*, abs/2404.16130, 2024. doi: 10.48550/ARXIV.2404.16130. URL https://doi.org/10.48550/arXiv.2404.16130.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997. doi: 10.1006/JCSS.1997.1504. URL https://doi.org/10.1006/jcss.1997.1504.

J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. doi: 10.1214/aos/1013203451. URL https://doi.org/10.1214/aos/1013203451.

L. Gao, X. Ma, J. Lin, and J. Callan. Precise zero-shot dense retrieval without relevance labels. In A. Rogers, J. L. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1762–1777. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.99. URL https://doi.org/10.18653/v1/2023.acl-long.99.

Z. Gou, Z. Shao, Y. Gong, Y. Shen, Y. Yang, N. Duan, and W. Chen. CRITIC: large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Sx038qxjek.

Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang. Lightrag: Simple and fast retrieval-augmented generation. In C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, editors, *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 10746–10761. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.findings-emnlp.568/.

B. J. Gutierrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su. Hipporag: Neurobiologically inspired long-term memory for large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/6ddc001d07ca4f319af96a3024f6dbd1-Abstract-Conference.html.

K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR, 2020. URL http://proceedings.mlr.press/v119/guu20a.html.

H. Han, Y. Wang, H. Shomer, K. Guo, J. Ding, Y. Lei, M. Halappanavar, R. A. Rossi, S. Mukherjee, X. Tang, Q. He, Z. Hua, B. Long, T. Zhao, N. Shah, A. Javari, Y. Xia, and J. Tang. Retrieval-augmented generation with graphs (graphrag). *CoRR*, abs/2501.00309, 2025. doi: 10.48550/ARXIV.2501.00309. URL https://doi.org/10.48550/arXiv.2501.00309.

X. Ho, A. D. Nguyen, S. Sugawara, and A. Aizawa. Constructing A multi-hop QA dataset for comprehensive evaluation of reasoning steps. In D. Scott, N. Bel, and C. Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 6609–6625. International Committee on Computational Linguistics, 2020. doi: 10.18653/V1/2020.COLING-MAIN.580. URL https://doi.org/10.18653/v1/2020.coling-main.580.

Y. Hu, Y. Zhao, C. Zhao, and A. Cohan. MCTS-RAG: enhancing retrieval-augmented generation with monte carlo tree search. In C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, editors, *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 12581–12597. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.findings-emnlp.672/.

G. Izacard and E. Grave. Leveraging passage retrieval with generative models for open domain question answering. In P. Merlo, J. Tiedemann, and R. Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 874–880. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.EACL-MAIN.74. URL https://doi.org/10.18653/v1/2021.eacl-main.74.

S. Jeong, J. Baek, S. Cho, S. J. Hwang, and J. Park. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In K. Duh, H. Gómez-Adorno, and S. Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 7036–7050. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.NAACL-LONG.389. URL https://doi.org/10.18653/v1/2024.naacl-long.389.

Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig. Active retrieval augmented generation. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7969–7992. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.495. URL https://doi.org/10.18653/v1/2023.emnlp-main.495.

S. Jiao, X. Xiao, Y. Wei, S. Qi, C. Huang, Q. Z. Sheng, and L. Yao. Prunerag: Confidence-guided query decomposition trees for efficient retrieval-augmented generation. In H. Hacid, Y. Maarek, F. Bonchi, I. Guy, and E. Yilmaz, editors, *Proceedings of the ACM Web Conference 2026, WWW 2026, Dubai, United Arab Emirates, originally scheduled for April 13-17, 2026, rescheduled for June 29 - July 3, 2026*, pages 1923–1934. ACM, 2026. doi: 10.1145/3774904.3792103. URL https://doi.org/10.1145/3774904.3792103.

J. Jin, Y. Zhu, Z. Dou, G. Dong, X. Yang, C. Zhang, T. Zhao, Z. Yang, and J. Wen. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In G. Long, M. Blumenstein, Y. Chang, L. Lewin-Eytan, Z. H. Huang, and E. Yom-Tov, editors, *Companion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*, pages 737–740. ACM, 2025. doi: 10.1145/3701716.3715313. URL https://doi.org/10.1145/3701716.3715313.

V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. Yih. Dense passage retrieval for open-domain question answering. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.550. URL https://doi.org/10.18653/v1/2020.emnlp-main.550.

G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3146–3154, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

O. Khattab, K. Santhanam, X. L. Li, D. Hall, P. Liang, C. Potts, and M. Zaharia. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *CoRR*, abs/2212.14024, 2022. doi: 10.48550/ARXIV.2212.14024. URL https://doi.org/10.48550/arXiv.2212.14024.

O. Khattab, A. Singhvi, P. Maheshwari, Z. Zhang, K. Santhanam, S. Vardhamanan, S. Haq, A. Sharma, T. T. Joshi, H. Moazam, H. Miller, M. Zaharia, and C. Potts. Dspy: Compiling declarative language model calls into state-of-the-art pipelines. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=sY5N0zY5Od.

T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=_nGgzQjzaRy.

P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

X. Li, G. Dong, J. Jin, Y. Zhang, Y. Zhou, Y. Zhu, P. Zhang, and Z. Dou. Search-o1: Agentic search-enhanced large reasoning models. In C. Christodoulopoulos, T. Chakraborty, C. Rose, and V. Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 5420–5438. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.276. URL https://doi.org/10.18653/v1/2025.emnlp-main.276.

A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Gupta, B. P. Majumder, K. Hermann, S. Welleck, A. Yazdanbakhsh, and P. Clark. Self-refine: Iterative refinement with self-feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

L. Mason, J. Baxter, P. L. Bartlett, and M. R. Frean. Boosting algorithms as gradient descent. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 512–518. The MIT Press, 1999. URL http://papers.nips.cc/paper/1766-boosting-algorithms-as-gradient-descent.

S. Min, V. Zhong, L. Zettlemoyer, and H. Hajishirzi. Multi-hop reading comprehension through question decomposition and rescoring. In A. Korhonen, D. R. Traum, and L. Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6097–6109. Association for Computational Linguistics, 2019. doi: 10.18653/V1/P19-1613. URL https://doi.org/10.18653/v1/p19-1613.

B. Peng, Y. Zhu, Y. Liu, X. Bo, H. Shi, C. Hong, Y. Zhang, and S. Tang. Graph retrieval-augmented generation: A survey. *ACM Trans. Inf. Syst.*, 44(2):35:1–35:52, 2026. doi: 10.1145/3777378. URL https://doi.org/10.1145/3777378.

E. Perez, P. Lewis, W. Yih, K. Cho, and D. Kiela. Unsupervised question decomposition for question answering. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8864–8880. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.713. URL https://doi.org/10.18653/v1/2020.emnlp-main.713.

O. Press, M. Zhang, S. Min, L. Schmidt, N. A. Smith, and M. Lewis. Measuring and narrowing the compositionality gap in language models. In H. Bouamor, J. Pino, and K. Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, Findings of ACL, pages 5687–5711. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EMNLP.378. URL https://doi.org/10.18653/v1/2023.findings-emnlp.378.

L. O. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin. Catboost: unbiased boosting with categorical features. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6639–6649, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/14491b756b3a51daac41c24863285549-Abstract.html.

R. Pryzant, D. Iter, J. Li, Y. T. Lee, C. Zhu, and M. Zeng. Automatic prompt optimization with "gradient descent" and beam search. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7957–7968. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.494. URL https://doi.org/10.18653/v1/2023.emnlp-main.494.

H. Qin, C. Wei, Y. Chen, and Y. Wang. Counterfactual reasoning for retrieval-augmented generation. In *The Fourteenth International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*. OpenReview.net, 2026. URL https://openreview.net/forum?id=9U51rOnGko.

A. Radhakrishnan, K. Nguyen, A. Chen, C. Chen, C. Denison, D. Hernandez, E. Durmus, E. Hubinger, J. Kernion, K. Lukosiute, N. Cheng, N. Joseph, N. Schiefer, O. Rausch, S. McCandlish, S. E. Showk, T. Lanham, T. Maxwell, V. Chandrasekaran, Z. Hatfield-Dodds, J. Kaplan, J. Brauner, S. R. Bowman, and E. Perez. Question decomposition improves the faithfulness of model-generated reasoning. *CoRR*, abs/2307.11768, 2023. doi: 10.48550/ARXIV.2307.11768. URL https://doi.org/10.48550/arXiv.2307.11768.

P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning. RAPTOR: recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=GN921JHCRw.

R. E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5:197–227, 1990. doi: 10.1007/BF00116037. URL https://doi.org/10.1007/BF00116037.

T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom. Toolformer: Language models can teach themselves to use tools. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Z. Shao, Y. Gong, Y. Shen, M. Huang, N. Duan, and W. Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In H. Bouamor, J. Pino, and K. Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, Findings of ACL, pages 9248–9274. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EMNLP.620. URL https://doi.org/10.18653/v1/2023.findings-emnlp.620.

N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao. Reflexion: language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

I. Stelmakh, Y. Luan, B. Dhingra, and M. Chang. ASQA: factoid questions meet long-form answers. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 8273–8288. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.EMNLP-MAIN.566. URL https://doi.org/10.18653/v1/2022.emnlp-main.566.

W. Su, Y. Tang, Q. Ai, Z. Wu, and Y. Liu. DRAGIN: dynamic retrieval augmented generation based on the real-time information needs of large language models. In L. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 12991–13013. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.702. URL https://doi.org/10.18653/v1/2024.acl-long.702.

H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal. Musique: Multihop questions via single-hop question composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554, 2022. doi: 10.1162/TACL_A_00475. URL https://doi.org/10.1162/tacl_a_00475.

H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In A. Rogers, J. L. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10014–10037. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.557. URL https://doi.org/10.18653/v1/2023.acl-long.557.

L. Wang, N. Yang, X. Huang, B. Jiao, L. Yang, D. Jiang, R. Majumder, and F. Wei. Text embeddings by weakly-supervised contrastive pre-training. *CoRR*, abs/2212.03533, 2022. doi: 10.48550/ARXIV.2212.03533. URL https://doi.org/10.48550/arXiv.2212.03533.

X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

X. Wang, Z. Wang, X. Gao, F. Zhang, Y. Wu, Z. Xu, T. Shi, Z. Wang, S. Li, Q. Qian, R. Yin, C. Lv, X. Zheng, and X. Huang. Searching for best practices in retrieval-augmented generation. In Y. Al-Onaizan, M. Bansal, and Y. Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 17716–17736. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.EMNLP-MAIN.981. URL https://doi.org/10.18653/v1/2024.emnlp-main.981.

C. Wei, H. Qin, S. He, Y. Wang, and Y. Chen. T-Retriever: tree-based hierarchical retrieval augmented generation for textual graphs. In *Fortieth AAAI Conference on Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 15924–15932. AAAI Press, 2026. doi: 10.1609/AAAI.V40I19.38625. URL https://doi.org/10.1609/aaai.v40i19.38625.

S. Yan, J. Gu, Y. Zhu, and Z. Ling. Corrective retrieval augmented generation. *CoRR*, abs/2401.15884, 2024. doi: 10.48550/ARXIV.2401.15884. URL https://doi.org/10.48550/arXiv.2401.15884.

Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics, 2018. doi: 10.18653/V1/D18-1259. URL https://doi.org/10.18653/v1/d18-1259.

S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023a. URL http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

T. Yu, S. Zhang, and Y. Feng. Auto-rag: Autonomous retrieval-augmented generation for large language models. *CoRR*, abs/2411.19443, 2024. doi: 10.48550/ARXIV.2411.19443. URL https://doi.org/10.48550/arXiv.2411.19443.

M. Yüksekgönül, F. Bianchi, J. Boen, S. Liu, P. Lu, Z. Huang, C. Guestrin, and J. Zou. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639(8055):609–616, 2025. doi: 10.1038/s41586-025-08661-4. URL https://doi.org/10.1038/s41586-025-08661-4.

E. Zelikman, Y. Wu, J. Mu, and N. D. Goodman. Star: Bootstrapping reasoning with reasoning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/639a9a172c044fbb64175b5fad42e9a5-Abstract-Conference.html.

H. S. Zheng, S. Mishra, X. Chen, H. Cheng, E. H. Chi, Q. V. Le, and D. Zhou. Take a step back: Evoking reasoning via abstraction in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=3bq3jsvcQ1.

D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. V. Le, and E. H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WZH7099tgfM.

A Finite-Time Termination and Zero Residual

We complement the monotone-descent guarantee of Theorem 5.1 with a finite-time termination result and a zero-residual characterization on the coordinates retained at an empty-active-set check.

Theorem A.1 (Finite-time termination of GBT-RAG)**.** *Under the assumptions of Theorem 5.1, Algorithm 1 terminates at a round $m^\star \le M$ and returns an answer $A_{m^\star}$ for which at least one of*

$$\begin{aligned} \mathcal{L}^{(m^\star)}(A_{m^\star}) < \varepsilon \quad \textit{or} \quad \mathcal{I}_{m^\star} = \emptyset \\ \textit{or} \quad \mathcal{L}^{(m^\star)}(A_{m^\star-1}) - \mathcal{L}^{(m^\star)}(A_{m^\star}) < \delta \quad \textit{or} \quad m^\star = M. \end{aligned} \tag{22}$$

holds. Moreover, in the case $\mathcal{I}_{m^\star} = \emptyset$, $A_{m^\star}$ is a descent fixed point *of the boosting recursion in the sense that*

$$\mathcal{L}^{(m^\star)}(A_{m^\star}) \;=\; 0 \qquad \textit{and} \qquad \mathcal{H}_{\mathcal{I}_{m^\star}} \;=\; \emptyset, \tag{23}$$

so no admissible weak learner can lower the loss below its current value.

Proof. Termination. The outer loop of Algorithm 1 executes at most M iterations, after which it returns A_M and the fourth predicate of (22) holds with $m^\star = M$. If the loop exits earlier, it does so through one of three return statements, respectively triggered by

$$\mathcal{L}^{(m)}(A_{m-1}) < \varepsilon, \qquad \mathcal{I}_m = \emptyset, \qquad \mathcal{L}^{(m)}(A_{m-1}) - \mathcal{L}^{(m)}(A_m) < \delta, \tag{24}$$

each of which matches one of the first three predicates of (22) at $m^\star = m$, where in the first case we use Theorem 5.1(17) to extend the bound from A_{m-1} to $A_{m^\star}$:

$$\mathcal{L}^{(m^\star)}(A_{m^\star}) \;\le\; \mathcal{L}^{(m^\star)}(A_{m^\star-1}) \;<\; \varepsilon. \tag{25}$$

Fixed-point characterization. Suppose $\mathcal{I}_{m^\star} = \emptyset$. By Eq. (11),

$$\big[\mathfrak{r}^{(m^\star)}\big]_\ell \;=\; \big(\tau - [\kappa^{(m^\star)}(A_{m^\star-1})]_\ell\big)_+ \;=\; 0, \qquad \ell \in [K_{m^\star}], \tag{26}$$

which entails $[\kappa^{(m^\star)}(A_{m^\star-1})]_\ell \ge \tau$ for every ℓ, hence

$$\mathcal{L}^{(m^\star)}(A_{m^\star-1}) \;=\; \sum_{\ell \in [K_{m^\star}]} w_\ell^{(m^\star)} \cdot 0 \;=\; 0. \tag{27}$$

Algorithm 1 returns $A_{m^\star-1}$ at the empty-active-set check before any rewrite is attempted; identifying $A_{m^\star}$ with this returned answer gives $\mathcal{L}^{(m^\star)}(A_{m^\star}) = 0$, the first equality in (23). The second equality follows from Eq. (12): the class of admissible weak learners is $\mathcal{H}_{\mathcal{I}_{m^\star}}$, defined as depth-two trees whose leaves correspond to sub-aspects in $\mathcal{I}_{m^\star}$, and an empty active set leaves no sub-aspects to populate the leaves, so $\mathcal{H}_{\mathcal{I}_{m^\star}} = \emptyset$. Any hypothetical candidate rewrite $\tilde{A}$ satisfies

$$\mathcal{L}^{(m^\star)}(\tilde{A}) \;\ge\; 0 \;=\; \mathcal{L}^{(m^\star)}(A_{m^\star}), \tag{28}$$

so the gate of Eq. (15) would reject every strict increase and admit only changes preserving $\mathcal{L}^{(m^\star)} = 0$. No residual-restricted learner can lower this retained-coordinate loss below zero; further coordinate expansion can still reveal missing facets. □

Remark. The four predicates of (22) are not mutually exclusive: budget exhaustion may coincide with stagnation, and $\mathcal{I}_{m^\star} = \emptyset$ implies $\mathcal{L}^{(m^\star)}(A_{m^\star}) = 0 < \varepsilon$. The disjunction is therefore a sound case analysis rather than a partition, and the fixed-point characterization (23) attaches strictly to the empty-active-set case, which is the only termination mode in which the hypothesis class itself is exhausted.

B Full Algorithm Pseudocode

This appendix presents the complete GBT-RAG procedure in three parts. Algorithm 1 gives the top-level boosting loop, which orchestrates the three phases. Algorithm 2 details Phase 0: the draft-and-critique pipeline that constructs the initial coordinate system $\Phi(T_0)$ and calibration weights $\mathbf{w}$. Figure 5 details the leaf-execution and bottom-up aggregation sub-procedure shared by all trees. Every numbered step cross-references the corresponding equation in the main paper. Table 4 summarises all notation used throughout.

Table 4: Notation reference.

Symbol	Meaning
q	Input query (token sequence)
$\mathcal{C} = \{d_1, \ldots, d_N\}$	Passage corpus
$\mathcal{R} : \mathcal{V}^* \to 2^{\mathcal{C}}$	Retriever oracle (top-k passages)
$\mathcal{M} : \mathcal{V}^* \to \mathcal{V}^*$	Language model oracle
$\phi : \mathcal{V}^* \to \mathbb{S}^{n-1}$	Residual encoder (unit sphere; default $n = 768$)
s_ℓ	Sub-aspect description of leaf ℓ
$\Phi(T) \in \mathbb{R}^{n \times K}$	Coordinate matrix $[\phi(s_1), \ldots, \phi(s_K)]$ (Eq. (2))
$\kappa_T(A) \in [-1, 1]^K$	Coverage map of answer A w.r.t. tree T (Eq. (3))
$\mathfrak{r}^{(m)} \in \mathbb{R}^{K_m}_{\geq 0}$	Text gradient (coverage residual) at round m (Eq. (4))
$\mathcal{L}^{(m)}(A)$	Coverage loss of A at round m (Eq. (4))
$\mathcal{I}_m \subseteq [K_m]$	Active set: leaves with positive residual
Δ_m	Evidence increment from residual tree T_m (Eq. (13))
$\mathcal{G}(A, \Delta, q)$	Length-capped, entity-preserving rewrite oracle (Eq. (14))
$\mathcal{E}^*(q, \mathcal{D}_q)$	Verified named-entity set for query q
$\tau \in (0, 1)$	Coverage threshold (margin parameter)
γ^*	Corpus-grounding similarity threshold
$\beta \in (0, 1)$	Deduplication threshold for new leaves
$\kappa_{\min}, \kappa_{\text{evid}}$	Dead-leaf pruning thresholds
ε, δ	Convergence and stagnation tolerances
M	Maximum boosting rounds (budget)
$K_{\max}$	Hard cap on total leaf count

Algorithm 1 GBT-RAG — Top-level boosting loop. Phase 0 builds the coordinate system; Phase 1 executes the initial tree; Phase 2 runs the residual boosting loop with a monotone-improvement gate. The four termination conditions guarantee finite-time termination (Theorem A.1).

Input: Query q; corpus $\mathcal{C}$; oracles $\mathcal{R}, \mathcal{M}, \phi$; hyperparameters $\tau, \gamma^*, \beta, \kappa_{\min}, \kappa_{\text{evid}}, \varepsilon, \delta, M, K_{\max}$
Output: Candidate answer $A \in \mathcal{V}^*$

[Phase 0] Construct the initial coordinate system.
$(\{s_\ell\}_{\ell=1}^{K_0}, \mathbf{w}) \leftarrow \textsc{Phase0}(q, \mathcal{R}, \mathcal{M}, \phi, \gamma^*)$ *Alg. 2; Eqs.* (5)–(6)
[Phase 1] Execute initial tree and obtain A_0.
$T_0 \leftarrow \textsc{BuildTree}(\{s_\ell\}, q)$ *depth-2 tree; dims* $= \mathbf{F}$
$A_0 \leftarrow \textsc{ExecuteAndAggregate}(T_0, \mathcal{R}, \mathcal{M})$ *Fig. 5; Eqs.* (7)–(8)
$\Phi^{(0)} \leftarrow \Phi(T_0)$; $\mathbf{w}^{(0)} \leftarrow \mathbf{w}$; $\mathcal{L}_{\text{prev}} \leftarrow \mathcal{L}^{(0)}(A_0)$
[Phase 2] Boosting loop.
for $m = 1$ **to** M **do**
  $(\Phi^{(m)}, \mathbf{w}^{(m)}) \leftarrow (\Phi^{(m-1)}, \mathbf{w}^{(m-1)})$
  // Adaptive coordinate expansion (Eqs. (9)–(10)*)*
  $\mathbf{F}_m^{\text{new}} \leftarrow \mathcal{M}(\pi_{\text{audit}}(A_{m-1}, \{s_\ell\}))$
  for each $s' \in \mathbf{F}_m^{\text{new}}$ **do**
    if $\max_\ell \langle \phi(s'), \phi(s_\ell) \rangle < \beta$ **and** $\gamma_{s'} \geq \gamma^*$ **and** $\#\,\text{cols}(\Phi^{(m)}) < K_{\max}$ **then**
      Append $\phi(s')$ with its identifier to $\Phi^{(m)}$; append $w_{s'}$ to $\mathbf{w}^{(m)}$
  // Text gradient and active set (Eq. (11)*)*
  $\mathfrak{r}^{(m)} \leftarrow (\tau \mathbf{1} - \kappa_{T^{(m)}}(A_{m-1}))_+$
  $\mathcal{I}_m \leftarrow \{\ell : [\mathfrak{r}^{(m)}]_\ell > 0\}$
  // Dead-leaf pruning
  for each $\ell \in \mathcal{I}_m$ **do**
    if $[\kappa(A_{m-1})]_\ell < \kappa_{\min}$ **and** $\max_{m' < m, \ell' \in T_{m'}} \langle \phi(e_{\ell'}), \phi(s_\ell) \rangle < \kappa_{\text{evid}}$ **then**
      Remove ℓ from $\Phi^{(m)}$, $\mathcal{I}_m$, $\mathfrak{r}^{(m)}$, and $\mathbf{w}^{(m)}$
  // Termination checks
  if $\mathcal{L}^{(m)}(A_{m-1}) < \varepsilon$ **then**
    return A_{m-1} (absolute convergence)
  if $\mathcal{I}_m = \emptyset$ **then**
    return A_{m-1} (no residual on retained leaves)
  // Residual tree construction (Eq. (12)*)*
  $T_m \leftarrow \textsc{BuildResidualTree}(\{s_\ell\}_{\ell \in \mathcal{I}_m}, A_{m-1}, q)$
  $\Delta_m \leftarrow \textsc{ExecuteAndAggregate}(T_m, \mathcal{R}, \mathcal{M})$
  // Composition and monotone gate (Eqs. (14)–(15)*)*
  $\tilde{A}_m \leftarrow \mathcal{G}(A_{m-1}, \Delta_m, q)$
  if $\mathcal{L}^{(m)}(\tilde{A}_m) \leq \mathcal{L}^{(m)}(A_{m-1})$ **then**
    $A_m \leftarrow \tilde{A}_m$ (accept)
    if $\mathcal{L}^{(m)}(A_m) < \varepsilon$ **then**
      return A_m
    if $\mathcal{L}^{(m)}(A_{m-1}) - \mathcal{L}^{(m)}(A_m) < \delta$ **then**
      return A_m (stagnation)
  else
    $A_m \leftarrow A_{m-1}$ (rollback)
  $\mathcal{L}_{\text{prev}} \leftarrow \mathcal{L}^{(m)}(A_m)$
return A_M (budget exhausted)

Algorithm 2 PHASE0 — Coordinate system construction via draft-and-critique. The draft (Step 0c) grounds the critique (Step 0d) in corpus evidence rather than abstract query structure. The corpus-grounding test (Step 0f) assigns calibration weights that prevent weakly-supported dimensions from dominating the coverage loss.

Input: Query q; retriever $\mathcal{R}$; LM $\mathcal{M}$; embedding ϕ; grounding threshold γ^*
Output: Leaf set $\{s_\ell\}_{\ell=1}^{K_0}$; weights $\mathbf{w} \in \mathbb{R}_{>0}^{K_0}$; verified entities $\mathcal{E}^*$

Step 0a — Initial retrieval.
$\mathcal{D}_q \leftarrow \mathrm{Top}_{10}(\mathcal{R}(q))$ *top-10 passages*
Step 0b — Named-entity verification.
Extract candidates via regex (capitalised phrases, quoted strings).
$\mathcal{E}^* \leftarrow \{e \in \mathcal{E}(q) : e \text{ appears verbatim in } \mathcal{D}_q\}$
Step 0c — Draft answer.
$\tilde{A} \leftarrow \mathcal{M}(\pi_{\mathrm{draft}}(q, \mathcal{D}_q))$ *2–3 sentence draft; Eq. 5*
Step 0d — Ambiguity critique.
$\mathbf{F} \leftarrow \mathcal{M}(\pi_{\mathrm{crit}}(q, \tilde{A}, \mathcal{D}_q, \mathcal{E}^*))$ *enumerate interpretations; Eq. 6*
$\mathbf{F} = \{(\mathrm{name}_d, \mathrm{expl}_d, S_d)\}_{d=1}^{D}, \quad S_d = \{s_{d,1}, \ldots, s_{d,|S_d|}\}$
if $|\mathbf{F}| < 2$ **and** no justification field present **then**
    Retry $\mathcal{M}$ once with stronger prompt; fall back to $\pi_{\mathrm{decompose}}$ if retry fails
Step 0e — Leaf set construction.
$\{s_\ell\}_{\ell=1}^{K_0} \leftarrow \bigsqcup_{d=1}^{D} S_d$ $K_0 = \sum_{d=1}^{D} |S_d|$
$\Phi(T_0) \leftarrow [\phi(s_1), \ldots, \phi(s_{K_0})] \in \mathbb{R}^{n \times K_0}$
Step 0f — Corpus-grounding test and weight assignment.
for each leaf $\ell \in [K_0]$ **do**
    $\gamma_\ell \leftarrow \max_{c \in \mathcal{R}(s_\ell)} \langle \phi(s_\ell), \phi(c) \rangle$ *best similarity; Eq. 7*
    if $\gamma_\ell \geq \gamma^*$ **then** $w_\ell \leftarrow 1.0$ **else** $w_\ell \leftarrow 0.5$
if fewer than 2 leaves have $w_\ell = 1.0$ **then**
    Re-admit highest-scoring weakly-grounded leaf at $w = 0.5$
return $\{s_\ell\}_{\ell=1}^{K_0}, \quad \mathbf{w} = (w_1, \ldots, w_{K_0}), \quad \mathcal{E}^*$

Remark B.1 (Termination, informal)**.** Algorithm 1 executes at most M boosting attempts. An accepted rewrite need not strictly lower loss or remove an active leaf. A rollback does not itself trigger the acceptance-only stagnation check. The finite cap guarantees termination independently of these behaviors; the possible earlier stopping conditions are described in Theorem A.1.

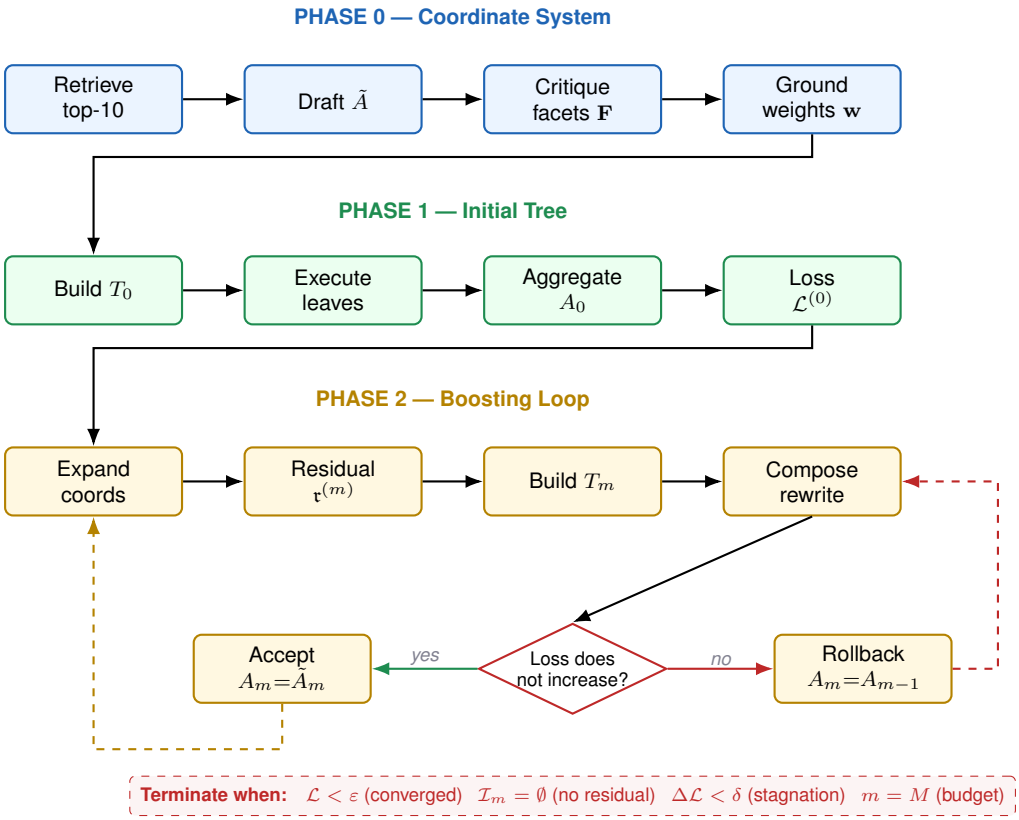

Figure 5: GBT-RAG phase flow diagram. Solid arrows denote forward data flow; dashed arrows denote the boosting loop-back to the next round. The four termination conditions (bottom) are reachable in finite time; see Theorem A.1.

C Hyperparameters

This appendix lists all hyperparameters of the GBT-RAG framework, their default values used to produce the main results, and the corresponding equation or section in which they are introduced.

Numeric defaults are shared across the three benchmarks (HotpotQA, MuSiQue, ASQA) and both backbones (Llama-3.1-8B-Instruct, Qwen2.5-7B-Instruct), while the Phase 0 prompt is task-dependent. The $D = 4$, $S = 3$ entries specify the generic `DECOMPOSE_QUERY` template. ASQA uses `AMBIGUITY_CRITIQUE`, which generates dimensions dynamically and can return different numbers of sub-aspects per dimension; its initial leaf count $K_0 = \sum_{d=1}^{D} |S_d|$ is not fixed to 12. The framework is robust within ± 1 DR in the local default regions tested, as established by the sensitivity analysis in Section 6.4.

D Benchmark and Data Details

This appendix expands on the three evaluation benchmarks introduced in §6.1. Each was selected to probe a distinct aspect of the *long-tailed coverage failure mode* that GBT-RAG addresses. Table 6 summarises key statistics at a glance; the subsections below give full details of construction, evaluation protocol, and how each benchmark stresses our framework.

D.1 HotpotQA [Yang et al., 2018]

Design rationale. HotpotQA is a Wikipedia-based multi-hop QA dataset comprising approximately 113,000 questions, each requiring *two-hop* reasoning over a pair of supporting

Table 5: Hyperparameter values used throughout the experiments.

Symbol	Description	Default	Reference
τ	Coverage margin (hinge loss threshold)	0.65	Eq. (4)
γ^*	Corpus grounding threshold	0.55	Eq. (6)
β	Deduplication threshold (coordinate expansion)	0.85	Eq. (10)
$\kappa_{\min}$	Dead-leaf coverage cutoff	0.15	Sec. 4.4, Step 2
κ_{evid}	Dead-leaf evidence cutoff	0.30	Sec. 4.4, Step 2
ε	Absolute convergence tolerance	10^{-2}	Eq. (19)
δ	Stagnation tolerance	10^{-3}	Eq. (20)
M	Maximum boosting rounds	3	Sec. 4.4
$K_{\max}$	Coordinate cap (maximum total leaves)	24	Sec. 4.4, Step 1
D	Dimensions in generic decomposition	4	Sec. 4.2
S	Sub-aspect prompt parameter	3	Sec. 4.2
k	Top-k retrieval depth	10	Sec. 4.2
n	Default residual dimension (`e5-base-v2`)	768	Sec. 3
Generation configuration			
Decoding	Greedy decoding	—	—
Max output tokens	Final-answer length cap	256	—

Table 6: **Dataset summary.** "Eval subset" is the randomly sampled evaluation partition used in all experiments (seed 42). Retrieval corpus for all datasets is the Dec-2018 DPR Wikipedia dump. ASQA evaluation size denotes 500 draws per sample with replacement from its 948-question pool.

Dataset	Hops	Dev size	Eval split	Eval subset	Answer type	Primary metric
HotpotQA	2	∼7,405	full-wiki dev	1000	Short span	Token F1
MuSiQue	2–4	∼2,417	answerable dev	1000	Short span	Token F1
ASQA	— / ambig.	948	dev	500	Long-form	Disambig-F1

documents. Questions are annotated with sentence-level supporting facts, enabling fine-grained evaluation of evidence chaining. We evaluate on the **open-domain (full-wiki) split**, where the system retrieves from the entire Wikipedia corpus rather than from a curated candidate set.

Data format. Each raw sample is a JSON object whose `"context"` field is a list of `(title, sentences[])` pairs. Sentences are joined into a single paragraph string per document. Our loader (`loader.py` : `load_hotpotqa`) reads `hotpot_dev_distractor_v1.json` and produces the internal schema shown in Table 10.

Why HotpotQA stresses GBT-RAG. HotpotQA primarily probes the framework's ability to chain *prominent* evidence: most questions have two readily-retrievable supporting passages. Gains over the strongest baselines on this benchmark therefore indicate improved coverage even on *shorter* reasoning chains, where iterative re-querying can in principle suffice but the structural residual provides additional precision.

Evaluation. We report `Token F1` and `Exact Match (EM)` using the SQuAD-style normalisation pipeline (lowercase → remove punctuation → remove articles → collapse whitespace) implemented identically to the official HotpotQA scorer `hotpot_evaluate_v1.py`.

D.2 MuSiQue [Trivedi et al., 2022]

Design rationale. MuSiQue (*Multi-hop Questions via Single-hop Composition*) constructs 2-, 3-, and 4-hop questions by composing chains of atomic single-hop questions. The construction

procedure deliberately resists the *disconnected-reasoning shortcuts* that plague HotpotQA-style datasets: each composed question can be answered correctly only by retrieving and integrating evidence at *every* hop, and disconnected partial reasoning is strongly penalised.

Data format. MuSiQue is distributed as a JSONL file (`musique_ans_v1.0_dev.jsonl`), one object per line. Our loader (`loader.py : load_musique`) retains only *answerable* samples (`"answerable" = true`); unanswerable entries are discarded before sampling. Each sample provides multiple surface-form `answer_aliases`, all of which are scored and the maximum taken—matching MuSiQue's official evaluation protocol.

Why MuSiQue stresses GBT-RAG. MuSiQue is the strongest test of whether the *residual signal* correctly localises which hop is missing, rather than re-litigating already-covered ones. The deeper hop counts (3- and 4-hop questions in particular) probe the long-tailed regime where iterative paraphrase-based methods saturate because they cannot distinguish which specific hop has zero coverage.

Hop count	2-hop	3-hop	4-hop
Fraction of dev set	∼50%	∼25%	∼25%

Table 7: Approximate hop-count distribution in the MuSiQue answerable dev split (see stratified analysis in §6.6).

Evaluation. We report `Token F1` and `Exact Match (EM)` with multi-alias scoring (`compute_score_with_aliases` in `evaluate.py`), taking the maximum over all surface-form aliases.

D.3 ASQA [Stelmakh et al., 2022]

Design rationale. ASQA (*Answer Summaries for Questions which are Ambiguous*) consists of approximately 6,000 ambiguous factoid questions drawn from AmbigQA, paired with long-form answers that explicitly enumerate *all* disambiguated interpretations the question admits. A question with two valid interpretations under different temporal or geographic disambiguations carries a gold answer covering both. This structure directly probes the long-tailed coverage failure mode: a system that commits to one interpretation, even correctly, loses credit for failing to cover the others.

Data format. The on-disk format (`data/asqa/dev.json`) is a list of sample objects. Each sample provides: a `question`, one or more long-form `annotations` (each a `long_answer` string), and a list of `qa_pairs` each carrying `short_answers` strings (the per-interpretation factoid answers used for Disambig-F1). Our loader (`loader.py : load_asqa`) extracts all long-form answers as `answer_aliases` and all short answers as a flat `short_answers` list for offline evaluation. Gold answers are not supplied to generation or to the Phase 7 compression guard; compression does not use evaluation labels.

Task type. ASQA samples are tagged `task_type = "longform"` in our internal schema, which activates the long-form generation path in GBT-RAG: short-answer distillation is skipped, Phase 0 uses the `AMBIGUITY_CRITIQUE` prompt (rather than `DECOMPOSE_QUERY`), with a query-dependent number of dimensions and sub-aspects; the resulting initial tree may contain more than 12 leaves. The generic four-dimension, three-sub-aspect template does not impose a 12-leaf cap on this path. For ASQA, the per-tree aggregation budget is set to 250 words, and Phase 7 format-aligned compression is enabled with a target of 65 words (matching the gold average; see `asqa.yaml`).

Evaluation. We report three complementary metrics:

- `Disambig-F1` QA-reader short-answer F1 averaged over gold disambiguated questions (Appendix F), rewarding coverage of the interpretations.

- `ROUGE-L` Official `rougeLsum` F-measure between prediction and reference long-form answer (`rouge_score` library, `use_stemmer=True`). Rewards lexical similarity to the gold answer.
- `DR score` Composite $\mathrm{DR} = \sqrt{\text{Disambig-F1} \times \text{ROUGE-L}}$, combining dataset-level disambiguation accuracy and reference similarity. This is the *tie-breaking* metric we use when Disambig-F1 and ROUGE-L move in opposite directions.

D.4 Corpus and Retrieval Infrastructure

All three benchmarks retrieve from the **Dec-2018 DPR Wikipedia dump** [Karpukhin et al., 2020], the standard open-domain RAG corpus. Retrieval uses `e5-base-v2`, which is also the default residual encoder. Retrieval vectors are unit-normalized and 768-dimensional. A **FAISS** `IndexFlatIP` index is used for exact inner-product (cosine) search. The cross-encoder reranker (`bge-reranker-v2-m3`) is applied at the leaf level when enabled.

Table 8 shows the retrieval configuration used for all experiments. Hyperparameters are fixed across datasets; the only dataset-specific change is the aggregation word budget for ASQA long-form outputs.

Table 8: Retrieval and embedding configuration used in all experiments.

Component	Setting	Value
Corpus	Wikipedia dump	Dec-2018 DPR
Retrieval encoder	`e5-base-v2`	$d = 768$
Index type	FAISS `IndexFlatIP`	Exact cosine search
Retrieval top-k	Per-leaf retrieval	$k = 10$
Reranker	`bge-reranker-v2-m3`	Cross-encoder, top-5
Retrieval method	Dense (default)	BM25 available (ablation)
Batch size	Embedding encoding	64

D.5 Evaluation Subsets and Sampling

Table 1 reports means over five evaluation-question samples, with greedy decoding and common samples for methods within each comparison. For ASQA, the sampling pool is the public development split of 948 questions. Each of the five samples consists of 500 draws with replacement from this pool. Thus, 500 is the number of draws, not necessarily the number of distinct questions. Seed 42 identifies the reference sample.

Table 9: Evaluation subset details. File paths correspond to the config files supplied with the code release. For ASQA, 500 denotes draws per sample with replacement from the 948-question pool; seed 42 denotes the reference sample.

Dataset	Source file	Dev size	Eval subset	Seed
HotpotQA	`hotpot_dev_distractor_v1.json`	7,405	1000	42
MuSiQue	`musique_ans_v1.0_dev.jsonl`	2,417*	1000	42
ASQA	`dev.json`	948	500	42

*Answerable samples only; the full MuSiQue dev set contains ∼4,000 entries.

E Baseline Descriptions

This appendix provides per-method descriptions and reproduction notes for the seven baselines compared in Table 1. The methods are organized by their position in the iterative/structural taxonomy of Section 1.

Table 10: Unified internal sample schema produced by all dataset loaders in `src/loader.py`. Fields marked † are dataset-specific and may be empty for other benchmarks.

Field	Type	Present in	Description
`id`	`str`	All	Dataset-assigned question ID
`question`	`str`	All	Question string
`answer`	`str`	All	Canonical gold answer
`answer_aliases`	`list[str]`	All	All valid surface forms (including canonical)
`short_answers`	`list[str]`	ASQA†	Per-interpretation factoid answers (Disambig-F1)
`context`	`list[dict]`	All	Candidate document pool `{title, text}`
`supporting_facts`	`list`	HotpotQA, MuSiQue†	Gold evidence document titles / paragraph indices
`task_type`	`str`	All	`"shortform"` or `"longform"`
`dataset`	`str`	All	Dataset identifier string

E.1 Sequential Baseline

Standard RAG [Karpukhin et al., 2020]. The canonical retrieve-then-read baseline. Given a question, the retriever returns the top-k passages, which are concatenated with the question and passed to the language model in a single decoding pass. This baseline carries no iteration, no decomposition, and no critique mechanism, and serves as the reference point against which all richer methods are measured. We reproduce Standard RAG with $k = 10$ passages, matching FlashRAG defaults.

E.2 Iterative Agentic Methods

FLARE [Jiang et al., 2023]. FLARE (Forward-Looking Active REtrieval) interleaves retrieval and generation by triggering re-retrieval whenever the language model's token-level confidence on a forthcoming sentence falls below a threshold. The framework attempts to look ahead during generation and proactively fetch evidence before producing low-confidence content. FLARE's iteration is reactive and unstructured: each trigger initiates a fresh retrieval against the original corpus distribution, with no mechanism for tracking which aspects have been covered.

Iter-Retgen [Shao et al., 2023]. Iter-Retgen runs a fixed number of retrieve-generate rounds in which each round's generated answer becomes the query for the next round's retrieval. The intuition is that intermediate generations refine what the system needs to retrieve next. The method is purely iterative without explicit decomposition: each round re-queries the corpus with a paraphrased version of the current best answer, deepening coverage of already-prominent aspects but unable to surface rare ones.

Auto-RAG [Yu et al., 2024]. Auto-RAG treats retrieval planning as an agentic decision-making process: the language model itself decides when to retrieve, what to retrieve, and when to terminate. It uses a fine-tuned planner that interleaves reasoning steps with retrieval calls based on the model's assessment of remaining information needs. The agentic policy provides flexibility but no internal structure to guide where the next retrieval should look.

FAIR-RAG [Asl et al., 2025]. FAIR-RAG (Faithful Adaptive Iterative Refinement) performs structured evidence assessment between retrieval rounds, identifying gaps in the current evidence and refilling them on the next round. It is among the strongest iterative baselines on multi-hop QA, achieving state-of-the-art results on HotpotQA among methods using Llama-3-8B at the time of writing. FAIR-RAG's gap-identification step is a form of textual critique without the embedding-space residual that GBT-RAG introduces; consequently its critique is not commensurable across rounds in the way our text gradient is.

E.3 Structural Methods

MCTS-RAG [Hu et al., 2025]. MCTS-RAG augments retrieval-augmented generation with Monte Carlo tree search over reasoning trajectories. The system explores multiple candidate reasoning paths via UCT-style selection, with retrieval calls embedded at expansion nodes. MCTS-RAG was originally proposed for mathematical reasoning; its application to multi-hop QA is non-standard. The hierarchy MCTS-RAG explores is a search tree over reasoning paths rather than a decomposition tree over query aspects, which means it inherits the corpus's retrieval bias at every node.

PruneRAG [Jiao et al., 2026]. PruneRAG constructs a confidence-guided query decomposition tree, expanding promising branches and pruning low-confidence ones in a single forward pass. It is the strongest structural baseline and a direct point of comparison for GBT-RAG: both methods use query decomposition trees, but PruneRAG commits to its hierarchy at $t = 0$ with no mechanism to admit aspects discovered during execution. PruneRAG thus exemplifies the "fixed hierarchy" failure mode identified in our introduction.

E.4 Reproduction Notes

For each baseline we use the implementation released by its authors when available, normalized to FlashRAG [Jin et al., 2025] reproducibility settings to eliminate confounding from incompatible retrievers, prompts, or decoding configurations. Several method-specific notes apply:

- **FAIR-RAG**: re-run on Llama-3.1-8B-Instruct (the original paper uses Llama-3-8B-Instruct); on Qwen2.5-7B-Instruct, run from the public implementation with default hyperparameters.
- **Self-RAG** [Asai et al., 2024] is omitted from the main comparison because its released checkpoint is fine-tuned on Llama-2-7B and the reflection-token vocabulary is not transferable to other backbones; running it on a different backbone requires retraining the critic, which is outside the scope of this work.
- **All other baselines** are run with default hyperparameters from their original papers, with retrieval normalized to top-10 against the FlashRAG-aligned corpus and decoding fixed to greedy with a 256-token cap.

F Evaluation Metrics

This appendix details the evaluation metrics reported in Section 6, with particular attention to the ASQA metrics, which are less standard than F1/EM.

Token-level F1. For HotpotQA and MuSiQue, F1 is computed at the token level following the SQuAD-style protocol used in both benchmarks' official evaluation scripts. Given a predicted answer $\hat{a}$ and a gold answer $a^\star$, both are tokenized and lowercased, with articles and punctuation removed and whitespace normalized. F1 is the harmonic mean of token precision and recall:

$$\mathrm{F1}(\hat{a}, a^\star) = \frac{2 \cdot |T(\hat{a}) \cap T(a^\star)|}{|T(\hat{a})| + |T(a^\star)|}, \tag{29}$$

where $T(\cdot)$ denotes the bag of normalized tokens. When multiple gold answers exist (HotpotQA does not; MuSiQue may), F1 is the maximum over the gold set.

Exact Match (EM). EM is a binary metric returning 1 if the normalized prediction string exactly equals a normalized gold answer and 0 otherwise. The same normalization (lowercasing, article and punctuation removal) is applied as for F1. EM is strictly more conservative than F1 and is sensitive to surface-form variation; we report both to distinguish content correctness (F1) from string match (EM).

ASQA Disambig-F1. Following the official ASQA evaluation pipeline, for each gold disambiguated question q_i, a separate QA reader extracts a short answer $\widehat{a}_i$ from the generated long answer $\hat{A}$. This is compared with acceptable aliases $\mathcal{Y}_i$:

$$\text{Disambig-F1}(\hat{A}, q) = \frac{1}{n} \sum_{i=1}^{n} \max_{a \in \mathcal{Y}_i} \mathrm{F1}(\widehat{a}_i, a). \tag{30}$$

The reported score averages over examples. It is not direct token overlap between the entire long answer and each short answer.

ASQA ROUGE-L. The official scorer uses the `rougeLsum` F-measure with stemming and sentence tokenization, selecting the better reference when two are available. This reference-similarity score is reported as ROUGE-L in the tables; it is not a direct fluency measurement.

ASQA DR (Disambig-ROUGE). DR is the geometric mean of the dataset-level aggregate scores [Stelmakh et al., 2022]:

$$\mathrm{DR} = \sqrt{\text{Disambig-F1} \cdot \text{ROUGE-L}}. \tag{31}$$

It combines disambiguated answer accuracy and reference similarity. Taking the geometric mean after aggregation is not interchangeable with averaging per-question geometric means.

Reporting conventions. All metrics are reported as percentages with one decimal place. Aggregate ASQA answer-quality scores are evaluated after final format-aligned compression. The loss and active-set diagnostics in Table 3 are computed on the uncompressed answers used by the boosting loop; cumulative retrieval counts describe that same loop. Table 1 reports the arithmetic mean of benchmark scores over five evaluation-question samples. ASQA scores are computed using the official evaluation pipeline for each sample before averaging across samples; the 500-draw, with-replacement sampling protocol is described in Appendix D.

G Case Study: A Complete Trace of GBT-RAG

To make the operation of GBT-RAG concrete, we provide a complete trace on the ASQA question *"Who sang the song I'm Coming Out?"*. This question exemplifies the long-tailed coverage failure mode that motivates our framework: the prominent answer (Diana Ross, 1980) dominates the corpus, while the gold answer also requires identifying Amerie's 2003 cover for the *Maid in Manhattan* soundtrack a rare interpretation unreachable by paraphrase-driven retrieval. Table 11 traces the algorithm phase by phase.

Table 11: Complete trace of GBT-RAG on the ASQA question *"Who sang the song I'm Coming Out?"* The gold short answers are {Diana Ross, Amerie}. Hyperparameters: $\tau = 0.65$, $\gamma^* = 0.55$, $\beta = 0.85$.

Stage	Output / State			
PHASE 0 — Coordinate System Construction				
Anchor retrieval & draft	$\mathcal{D}_q$: top-10 passages (Diana-Ross-dominated). $\tilde{A}$: *"... written by Bernard Edwards and Nile Rodgers, recorded by Diana Ross in 1980 ..."*			
Entity extraction	$\mathcal{E}^* = \{$*I'm Coming Out, Diana Ross, Bernard Edwards, Nile Rodgers, 1980*$\}$			
Dimensions & grounding	**Dimension**	γ_ℓ	w_ℓ	$K_0 = 6$ leaves; marginal facet weighted at 0.5 via Eq. (7).
	Diana Ross Version	0.81	1.0	
	Manhattan Transfer Version	0.62	1.0	
	Cover Versions and Samples	0.74	1.0	
	Cultural Significance	0.58	1.0	
	Personal Significance to Ross	0.71	1.0	
	Other Interpretations	0.41	**0.5**	
PHASE 1 — Initial Learner T_0 and Answer A_0				

Continued on next page

Table 11 – *Continued from previous page*

Stage	Output / State

A_0 (after aggregation): *"I'm Coming Out is a song recorded by Diana Ross in 1980, written by Bernard Edwards and Nile Rodgers … became an LGBT anthem … The Manhattan Transfer recorded a version, and The Notorious B.I.G. sampled it … "*

Coverage & residual $r^{(0)}$:

Leaf	$\kappa(A_0)$	$r_\ell^{(0)}$
1. Diana Ross Version	0.73	0.00
2. Manhattan Transfer	0.62	0.08
3. Covers and Samples	0.72	0.00
4. Cultural Significance	0.67	0.03
5. Personal Significance	0.72	0.00
6. Other Interpretations	0.57	0.13

$\mathcal{L}^{(0)} = 0.395$, **Disambig-F1** = 33.3

PHASE 2, ROUND 1 — Adaptive Expansion, Sparse Tree, Rewrite

Audit candidates ($\Phi^{(1)}$):

Candidate	$\max\langle\phi, \phi_\ell\rangle$	γ	Test	Verdict
Amerie's 2003 cover	0.43	0.68	$< \beta, \geq \gamma^*$	**Admit**
Keyshia Cole remake	—	0.39	$< \gamma^*$	Reject
LGBT anthem reception	0.81	—	$> \beta$ (dup.)	Reject

extends to $K_1 = 7$ leaves.

Active set & sparse tree: $\mathcal{I}_1 = \{2, 4, 6, \mathbf{7}\}$, with $r_7^{(1)} = \mathbf{0.39}$ (largest residual). T_1 is grown over only these 4 leaves; the 3 covered leaves are skipped.

Targeted retrieval: q_7: *"Amerie I'm Coming Out cover Maid in Manhattan soundtrack"*

e_7: *"In 2003, American singer Amerie covered 'I'm Coming Out' for the soundtrack to the film Maid in Manhattan … "*

Composition & gate: $\mathcal{L}^{(1)}(\tilde{A}_1) = 0.211 \leq \mathcal{L}^{(1)}(A_0) = 0.527 \quad \Rightarrow \quad$ **ACCEPT** $\quad A_1 \leftarrow \tilde{A}_1$

PHASE 2, ROUND 2 — Rollback

Active set & rewrite: $\mathcal{I}_2 = \{4, 6\}$. Rewrite introduces tangential material on Beyoncé's 2007 sampling, diluting the entity-preserving structure.

Gate decision: $\mathcal{L}^{(2)}(\tilde{A}_2) = 0.264 > \mathcal{L}^{(2)}(A_1) = 0.211 \quad \Rightarrow \quad$ **ROLLBACK** $\quad A_2 \leftarrow A_1$

PHASE 2, ROUND 3 — Termination

State: $\mathcal{I}_3$ unchanged; audit yields no new admissible candidates; stagnation criterion (Eq. 20) triggers. **Final**: $A_3 = A_1$.

Convergence Trace

Round	$\mathcal{L}^{(m)}$	$\lvert\mathcal{I}_m\rvert$	Cum. retr.	Decision	Disambig-F1
0 (Phase 1)	0.395	3	6	—	33.3
1	**0.211**	4	10	Accept	**66.7**
2	0.264 (cand.)	2	12	Rollback	66.7
3	0.211	0	12	Terminate	66.7

Final Answer (after compression)

"I'm Coming Out is a 1980 song recorded by Diana Ross, written and produced by Chic members Bernard Edwards and Nile Rodgers … In 2003, American singer Amerie covered the song for the soundtrack to the film Maid in Manhattan … "

H Discussion of the Gradient Boosting Analogy

H.1 Where the Analogy Holds, and Where It Breaks

The framing of GBT-RAG as gradient boosting in text space rests on a structural analogy with the three commitments (C1)–(C3) of Friedman [2001]. We made this correspondence concrete in Sections 4–5: the coverage residual plays the role of the negative functional gradient, decomposition trees play the role of weak learners, and the gated rewrite plays the role of additive composition. We now make explicit which parts of the analogy are tight and which are loose, so that the contribution is not overstated.

Tight correspondences.

(i) The residual $\mathfrak{r}_T(A)$ is genuinely a per-coordinate nonnegative deficiency that vanishes at coordinates exceeding the margin and grows linearly below it; it is not the negative derivative of the stated hinge loss. Away from the hinge, that derivative is $w_\ell \mathbf{1}\{c_\ell < \tau\}$ for $c_\ell = [\kappa_T(A)]_\ell$; the residual instead retains the deficit magnitude.

(ii) Successive learners are fit only over the support of the residual (the active set $\mathcal{I}_m$), which mirrors the boosting prescription that each h_m should target current errors rather than re-fit the whole problem.

(iii) Composition admits each new learner only when the current-round aggregate loss does not increase.

Loose correspondences.

(i) Eq. (12) constructs the tree directly from active leaves rather than solving a numerical regression problem. The "fitting" step is therefore better understood as *evidence collection guided by the residual support* than as functional regression.

(ii) GBT-RAG has no learning rate ν_m: $\mathcal{G}$ either accepts the full update or rolls back. A continuous-strength analogue would require a notion of partial integration of Δ_m into A_{m-1}, which we leave to future work.

(iii) Theorem 5.1 establishes monotone descent, but this descent is enforced by the gate (Eq. 15), not produced by the geometry of the learner. We therefore do not claim the convergence-rate guarantees of classical GBT (e.g., quadratic loss bounds under bounded-gradient learners).

The contribution of GBT-RAG is thus better characterized as *transposing the structural skeleton of gradient boosting*—residual, learner over support, gated composition—into a domain where the underlying machinery (differentiation, regression) is unavailable. The analogy is generative for design but is not a literal embedding of GBT into text space, and we do not claim it is.

H.2 Empirical Diagnostics for Assumption (A2)

Theorem 5.1's cross-round monotonicity (18) requires that $\mathcal{G}$ be evidence-monotone: $\mathcal{P}(A_{m-1}) \subseteq \mathcal{P}(\tilde{A}_m)$, where $\mathcal{P}(\cdot)$ denotes the embedded segment cloud. For a black-box LLM rewrite under length caps, this assumption is unlikely to hold strictly. We measure proxies for evidence preservation on sampled ASQA rewrites.

Segment-retention proxy. On randomly sampled rewrites from the ASQA development set, we segment A_{m-1} and $\tilde{A}_m$ at the sentence level and measure the fraction of segments $p \in \mathcal{P}(A_{m-1})$ for which there exists $p' \in \mathcal{P}(\tilde{A}_m)$ with $\langle \phi(p), \phi(p') \rangle \geq 0.95$ (i.e., near-verbatim retention). The mean retention rate is $\mathbf{74.6}\%$, with a long lower tail driven by length-cap-induced compressions.

Coordinate-wise approximation test. Since the loss only depends on $\mathcal{P}$ through the maxima in Eq. (3), strict cloud inclusion is sufficient but not necessary. We instead measure, for every pre-existing leaf $\ell \in [K_{m-1}]$, whether

$$[\kappa^{(m)}(\tilde{A}_m)]_\ell \;\geq\; [\kappa^{(m)}(A_{m-1})]_\ell - \eta \tag{32}$$

for $\eta = 0.02$. This is an *approximate coordinate-wise preservation* test; the exact loss bound requires non-degradation without tolerance. The fraction of (rewrite, leaf) pairs satisfying this relaxed condition is $\mathbf{91.8}\%$, reflecting that even when the LLM rewrite is not strictly evidence-additive at the segment level, it preserves coverage on previously-covered leaves in the overwhelming majority of cases. The remaining 8.2% of rewrite–leaf pairs are not equivalent to rollback rounds: the aggregate gate can accept compensating gains and losses across leaves.

Gate catches residual violations. By construction, every accepted round satisfies $\mathcal{L}^{(m)}(\tilde{A}_m) \leq \mathcal{L}^{(m)}(A_{m-1})$ on the round-m coordinate system, so any (A2) violation severe enough to increase the aggregate loss is caught and rolled back. The empirical rollback rate of 24.7% is thus a direct measurement of how often the LLM rewrite violates the descent property in practice. Theorem 5.1 should therefore be read as a clean characterization of the gating discipline, with (A2) as a sufficient but not necessary condition for cross-round descent in practice. Table 12 summarizes the four measurements.

Table 12: Evidence-preservation diagnostics on 500 ASQA evaluation questions (Llama-3.1-8B-Instruct).

Measurement	**Rate (%)**
Gate acceptance rate (non-increasing aggregate loss)	75.3
Mean segment retention ($\langle \phi(p), \phi(p') \rangle \geq 0.95$)	74.6
Approximate per-leaf preservation ($\eta = 0.02$)	91.8
Gate-rollback rate (aggregate loss increases)	24.7

Relation to textual feedback. TextGrad [Yüksekgönül et al., 2025] and ProTeGi [Pryzant et al., 2023] can use numerical evaluation objectives. GBT-RAG differs in retaining explicit per-leaf numerical deficits to target retrieval and evaluate subsequent answers; the distinction is not the absence of numerical evaluation in textual-feedback methods.

I Compute and Latency Analysis

I.1 Setup

We measure three cost dimensions for every method in Table 1: (i) total LLM input tokens, (ii) total LLM output tokens, (iii) number of retrieval calls, all averaged per question across the evaluated ASQA question set with Llama-3.1-8B-Instruct. Wall-clock latency is measured on a single NVIDIA A100-80GB GPU with batch size 1 and excludes corpus-loading time. All baselines are run with the same retriever, corpus, and decoding configuration as in the main experiments. Token counts are obtained directly from the model tokenizer; retrieval calls include both initial and iterative retrievals.

I.2 Per-Question Cost Breakdown

Table 13 reports per-question average compute and latency for all methods evaluated in Table 1.

GBT-RAG consumes approximately $1.66\times$ the input tokens and $1.87\times$ the output tokens of the strongest structural baseline (PruneRAG), at $2.2\times$ the wall-clock latency. We therefore conduct a Pareto analysis (Section I.3) and a cost-matched comparison (Section I.4) to determine whether the quality gain is attributable to the additional compute or to the structural design.

I.3 Pareto Analysis

Figure 6 plots DR against total LLM token cost (input + output) on a logarithmic axis. GBT-RAG sits on the upper-right Pareto frontier at $(30{,}292 \text{ tokens},\ 38.2 \text{ DR})$. The closest competitor on the frontier is PruneRAG at $(18{,}136 \text{ tokens},\ 33.9 \text{ DR})$; the slope between these two points is 0.35 DR per kilo-token of additional compute. No default baseline setting shown here achieves DR above 34; the oracle restart control below reaches 35.1.

Table 13: Per-question compute and latency on ASQA (Llama-3.1-8B-Instruct, single A100-80GB, batch size 1). Token counts are summed across all LLM calls. DR is reproduced from Table 1 for reference.

Type	Method	LLM in	LLM out	Retrievals	Latency (s)	DR
Sequential	Standard RAG	2,140	118	1	2.4	30.1
Iterative	FLARE	6,820	264	3.2	7.8	31.4
	Iter-Retgen	6,450	312	3.0	8.1	29.3
	Auto-RAG	12,980	408	5.4	14.6	29.1
	FAIR-RAG	11,730	486	5.1	13.9	30.3
Structural	MCTS-RAG	24,630	802	11.7	28.4	31.5
	PruneRAG	17,540	596	9.3	21.5	33.9
Ours	**GBT-RAG**	**29,180**	**1,112**	**25.0**	**47.2**	**38.2**

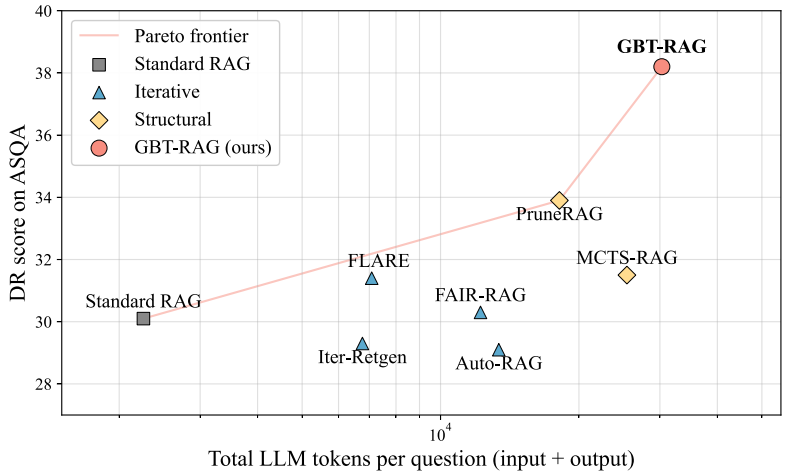

Figure 6: Cost–quality Pareto frontier on ASQA (Llama-3.1-8B-Instruct). GBT-RAG sits on the upper-right of the frontier at a regime no baseline reaches.

I.4 Cost-Matched Baseline Comparison

To isolate the contribution of GBT-RAG's structural design from raw compute spend, we run two expanded-budget controls.

Control 1: FAIR-RAG with extended budget. We re-run FAIR-RAG with its iteration cap raised from its default of 5 to 12, until its average per-question token cost (31,840 input tokens) approximately matches GBT-RAG's. Result: DR $= 32.4$, compared to 38.2 for GBT-RAG. The gap of 5.8 DR points shows that this expanded-budget FAIR-RAG configuration does not close the reported quality gap.

Control 2: PruneRAG with multi-restart oracle. PruneRAG is restarted with 3 independent decompositions (different temperature seeds) and the best-DR answer is selected by an oracle that has access to gold answers. Even with this favorable selection, DR $= 35.1$, still 3.1 DR below GBT-RAG. This tested restart strategy does not close the reported gap.

J Self-RAG Comparison on Aligned Backbone

To address the omission of Self-RAG [Asai et al., 2024] from the main comparison (Section 6), we report Self-RAG's published numbers on Llama-2-7B alongside our re-implementation of GBT-RAG on Llama-2-7B. We chose Llama-2-7B because the released Self-RAG checkpoint is fine-tuned on this backbone with reflection tokens not transferable to other backbones; running Self-RAG on

Table 14: Cost-matched controls. Both controls consume comparable or greater compute than GBT-RAG yet underperform.

Configuration	LLM in	Latency (s)	DR
PruneRAG (default)	17,540	21.5	33.9
PruneRAG (3× multi-restart, oracle select)	52,620	64.5	35.1
FAIR-RAG (default, cap=5)	11,730	13.9	30.3
FAIR-RAG (extended, cap=12)	31,840	41.2	32.4
GBT-RAG (ours)	**29,180**	**47.2**	**38.2**

Llama-3.1-8B-Instruct would require retraining the critic from scratch, which is outside the scope of this work.

We caution that this comparison is intended as a sanity check rather than a primary headline result: Llama-2-7B is a substantially weaker reasoner than the Llama-3.1-8B-Instruct and Qwen2.5-7B-Instruct backbones used in Table 1, and absolute scores for both methods are correspondingly lower. The relative gap between methods is nonetheless informative.

Table 15: Self-RAG comparison on the Llama-2-7B backbone. Self-RAG numbers are reproduced from Asai et al. [2024]; GBT-RAG numbers are from our re-implementation on Llama-2-7B with hyperparameters fixed at the defaults of Appendix C.

Method	HotpotQA		ASQA			MusiQue	
	F1	EM	DR	ROUGE-L	Disambig	F1	EM
Standard RAG	27.7	19.6	26.3	23.3	29.6	9.4	4.1
Self-RAG	35.6	27.4	30.6	27.1	34.5	15.7	7.9
GBT-RAG (ours)	**50.4**	**41.1**	**33.5**	**26.6**	**42.3**	**23.8**	**14.1**

GBT-RAG outperforms Self-RAG by $+19.2$ F1 on HotpotQA and $+13.2$ Disambig on ASQA at matched backbone, with the relative ordering consistent with the Llama-3.1-8B-Instruct results in Table 1. Self-RAG's reflection-token mechanism, while effective relative to Standard RAG, lacks both the structural decomposition of GBT-RAG's coordinate system and the localized residual signal of the text gradient; this manifests most sharply on the multi-hop and multi-interpretation regimes where coverage gaps require explicit localization rather than reflection-driven re-retrieval.

K Full Prompt Templates

This appendix reproduces every prompt template used in GBT-RAG verbatim. All templates are processed by a lightweight `fmt()` helper (described in Section K.15) that fills `{slot}`-style placeholders and fails gracefully on missing keys. Every prompt that requests structured data demands **JSON-only** output with no prose preamble, no markdown fences, and no trailing commentary; the system turn states this contract explicitly.

Notation. Highlighted tokens `{like_this}` are `fmt()`-filled slots. Badge colours indicate turn role: **SYSTEM** (system instruction) and **USER** (user turn). **OUTPUT SCHEMA** boxes show the expected output schema.

K.1 DECOMPOSE_QUERY

Purpose. Decomposes the original query into D top-level dimensions (interpretations, entities, aspects, or time periods), each with S sub-aspects. The output initialises the leaf coordinate system $\Phi(T_0)$.

SYSTEM

```
You are a meticulous research assistant that decomposes complex questions into
exhaustive, non-overlapping dimensions.
Rules:
1. Return ONLY valid JSON - no prose, no markdown fences.
2. Produce exactly {num_dimensions} dimensions.
3. Each dimension must have exactly {num_sub_aspects} sub_aspects.
4. sub_aspects must be atomic: one retrievable fact per entry.
5. Do NOT repeat information across dimensions.
6. Restrict all named entities to the verified set:
{verified_entities}
```

USER

```
Query: {query}
Draft answer (for context only - do not copy it):
{draft_answer}
Retrieved passages (for grounding):
{retrieved_passages}
Decompose the query into dimensions and sub-aspects as instructed.
```

OUTPUT SCHEMA

```
{
  "dimensions": [
    {
      "name": "<short label>",
      "ideal": "<one-sentence target>",
      "sub_aspects": ["<aspect 1>", "<aspect 2>", ...]
    }
  ]
}
```

K.2 AMBIGUITY_CRITIQUE

Purpose. Draft-and-critique step of Phase 0. Given the draft answer and retrieved documents, the model enumerates all distinct interpretations or facets that the draft omits.

SYSTEM

```
You are an expert critic tasked with identifying gaps in a draft answer.
Rules:
1. Return ONLY valid JSON - no prose, no markdown fences.
2. List every distinct interpretation, entity, or aspect that the draft MISSES
or covers insufficiently.
3. For each gap, provide 2-{num_sub_aspects} sub_aspects that a complete
answer would need.
4. Do NOT repeat anything the draft already covers well.
5. Restrict named entities to: {verified_entities}
```

USER

```
Original query:
{query}

Draft answer:
{draft_answer}

Retrieved passages:
{retrieved_passages}

List all interpretations or facets the draft misses.
```

OUTPUT SCHEMA

```
[
  {
    "name":  "<gap label>",
    "explanation":  "<why it is missing>",
    "sub_aspects":  ["<aspect 1>", "<aspect 2>", ...]
  },
  ...
]
```

K.3 SPLIT_QUESTION

Purpose. At an internal tree node, decomposes a sub-question into 2–4 child sub-questions, each more specific and independently answerable.

SYSTEM

```
You are a question-decomposition expert.  Given a sub-question, split it into
2-4 child sub-questions that together fully cover it.

Rules:
1. Return ONLY valid JSON.
2. Each child must be independently answerable via a single retrieval.
3. Children must be mutually non-overlapping.
4. Keep each child to one sentence.
```

USER

```
Parent question:  {sub_question}

Context (ancestor questions already answered):
{ancestor_context}

Split into 2-4 child sub-questions.
```

OUTPUT SCHEMA

```
{ "sub_questions":  ["<child 1>", "<child 2>", ...]  }
```

K.4 LEAF_DECISION

Purpose. Decides, for a node at depth d, whether it should remain a leaf (answer directly) or split further.

SYSTEM

```
You decide whether a question node is simple enough to answer directly (leaf) or
requires further decomposition (split).

Rules:
1. Return ONLY valid JSON.
2. A node is a leaf if it can be answered by retrieving a single passage.
3. A node should split if it contains multiple independent facts.
4. Maximum tree depth is {max_depth}.
```

USER

```
Current depth: {current_depth}

Node question: {sub_question}

Should this node be a leaf or split?
```

OUTPUT SCHEMA

```
{ "is_leaf": true | false }
```

K.5 MAKE_SEARCH_QUERY

Purpose. Operator $\mathcal{Q}$: converts a leaf sub-aspect description into a concise 3–10-word keyword query suitable for BM25 / dense retrieval.

SYSTEM

```
You convert a sub-aspect description into a short retrieval query.

Rules:
1. Return ONLY valid JSON.
2. The query must be 3-10 words.
3. Use only keywords - no question words, no filler.
4. Preserve all named entities verbatim.
```

USER

```
Sub-aspect: {sub_aspect}

Write a 3-10-word retrieval query.
```

OUTPUT SCHEMA

```
{ "query": "<keyword query>" }
```

K.6 EXTRACT_EVIDENCE

Purpose. Operator $\mathcal{X}$: extracts a verbatim evidence string from the top-k retrieved passages for a leaf node. Evidence is passed upward during bottom-up aggregation.

SYSTEM

```
You are a precise evidence extractor. Given a sub-aspect and retrieved
passages, locate and return only the text that directly supports the sub-aspect.

Rules:
```

```
1. Return the evidence as plain text (no JSON).
2. Copy text verbatim - do not paraphrase.
3. If multiple passages are relevant, concatenate the relevant spans with "
[...]  " between them.
4. If no passage supports the sub-aspect, return the single token:  NONE.
5. Preserve all named entities, dates, and numbers exactly.
```

USER

```
Sub-aspect:  {sub_aspect}

Retrieved passages:
{retrieved_passages}

Extract verbatim evidence for the sub-aspect.
```

K.7 AGGREGATE

Purpose. Bottom-up aggregation operators $\mathcal{A}_{\text{dim}}$ and $\mathcal{A}_{\text{root}}$. The same template is reused at both levels; the `{level}` slot switches behaviour.

SYSTEM

```
You are an expert writer that synthesises evidence into a coherent answer.  You
are operating at the {level} level (dimension | root).

Rules:
1. Return plain text only.
2. Do NOT paraphrase named entities, dates, or numerical quantities.
3. Integrate all provided evidence; do not invent unsupported claims.
4. dimension level:  write 2-4 sentences summarising this dimension.
5. root level:  write a complete, well-structured answer of at most
{max_words} words covering all dimensions.

6. Preserve the verified entity set:  {verified_entities}
```

USER

```
Query:  {query}

[dimension level]  Dimension:  {dimension_name}
Evidence snippets:
{evidence_list}

[root level]  Dimension summaries:
{dimension_summaries}
```

K.8 DISCOVER_DIMENSIONS

Purpose. Audit operator: proposes new candidate facets absent from the current leaf set. Each candidate is subsequently subject to the deduplication and grounding tests of Eq.(10).

SYSTEM

```
You are a coverage auditor. Given the current answer and the existing leaf
descriptions, propose facets that are MISSING from the answer and NOT already
represented by any existing leaf.

Rules:
1. Return ONLY valid JSON.
2. Propose at most {max_new_dims} new facets.
3. Each facet must be semantically distinct from all existing leaves (cosine
similarity < {beta_threshold} in embedding space).
4. Only propose facets that a keyword search can plausibly retrieve.
5. If no facet is missing, return an empty list: [].
```

USER

```
Query: {query}

Current answer:
{current_answer}

Existing leaf descriptions (already covered):
{existing_leaves}

Propose new facets not yet covered.
```

OUTPUT SCHEMA

```
[
  {
    "name": "<facet label>",
    "explanation": "<why this is missing>",
    "sub_aspects": ["<aspect 1>", ...]
  },
  ...
]
```

K.9 SUGGEST_BETTER_QUERY

Purpose. Fallback re-querying for a gap dimension whose initial retrieval returned NONE or below-threshold evidence.

SYSTEM

```
You suggest an alternative retrieval query for a gap facet that returned no
useful evidence on the first attempt.

Rules:
1. Return ONLY valid JSON.
2. The new query must be 3-10 words.
3. Use different vocabulary than the original query.
4. Do not introduce unverified named entities.
```

USER

```
Gap facet: {gap_description}

Original query: {original_query}
Failed passages (returned but unhelpful):
{failed_passages}
```

```
Suggest an alternative keyword search query.
```

OUTPUT SCHEMA

```
{ "query": "<alternative keyword query>" }
```

K.10 REWRITE and REWRITE_ANCHORED

Purpose. Composition operators $\mathcal{G}$. REWRITE produces a full rewrite that integrates residual evidence; REWRITE_ANCHORED is an extend-only variant (1–3 new sentences) used when rollback risk is higher.

REWRITE

SYSTEM

```
You are a careful writer that improves an answer by incorporating new evidence
for previously uncovered aspects.

Rules:
1. Return plain text only.
2. Preserve ALL named entities, dates, and numerical values from the previous
answer exactly as written.
3. The new answer must cover everything in the previous answer PLUS the new
evidence.
4. Maximum length: {max_words} words.
5. Do not invent claims not supported by evidence.
6. Verified entity set (must not be altered): {verified_entities}
```

USER

```
Query: {query}

Previous answer:
{previous_answer}

New evidence for uncovered aspects:
{new_evidence}

Write an improved answer that incorporates the new evidence.
```

REWRITE_ANCHORED

SYSTEM

```
You extend an existing answer by adding 1-3 sentences that cover a newly
retrieved aspect. You must NOT modify any existing sentence.

Rules:
1. Return plain text only.
2. Append 1-3 new sentences at the end of the existing answer.
3. Do not delete, reorder, or alter any existing sentence.
4. New sentences must be grounded solely in the provided evidence.
5. Preserve all named entities verbatim.
```

USER

```
Query: {query}

Existing answer (DO NOT MODIFY):
{previous_answer}

New evidence (append coverage for this aspect):
{new_evidence}
```

K.11 COMPOSE

Purpose. Additive composition operator used in the ablation study (Section 5.2) to isolate the effect of the monotone-gated rewrite. Unlike REWRITE, this operator simply appends a new paragraph without checking coherence or redundancy.

SYSTEM

```
You append a brief paragraph to an existing answer to cover a missing aspect.
Do not alter the existing text.

Rules:
1. Return plain text only.
2. Append exactly one paragraph of 2-4 sentences.
3. The paragraph must be grounded in the provided evidence.
4. Do not repeat information already in the answer.
```

USER

```
Existing answer:
{previous_answer}

Missing aspect: {gap_description}

Evidence: {evidence}
```

K.12 EXTRACT_SHORT_ANSWER

Purpose. Distils a detailed long-form answer into a short span for evaluation on HotpotQA and MuSiQue, where token-level F1/EM against reference spans is the metric.

SYSTEM

```
You distil a detailed answer into the shortest phrase that directly answers the
question, suitable for token-level F1 evaluation.

Rules:
1. Return plain text only (no JSON, no punctuation at end).
2. The answer must be a verbatim span or minimal paraphrase of content in the
detailed answer.
3. Maximum 10 words.
4. If the answer is a list, join items with ", ".
5. Preserve named entities and numbers exactly.
```

USER

```
Question: {query}

Detailed answer:
```

```
{detailed_answer}

Retrieved evidence (for verification):
{retrieved_passages}

Return the shortest correct answer span.
```

K.13 STANDARD_RAG, CRITIQUE, REFINE

Purpose. Baseline prompts used by Standard RAG and the Iterative RAG variants (FLARE, Iter-Retgen) in our comparison. These are reproduced here to ensure reproducibility of the baseline results in Table 1.

STANDARD_RAG

SYSTEM

```
You are a knowledgeable assistant.  Answer the question using only the provided
passages.  Be concise and factual.

Rules:
1. Return plain text only.
2. Do not add information not present in the passages.
3. Maximum {max_words} words.
```

USER

```
Question:  {query}

Passages:
{retrieved_passages}

Answer the question.
```

CRITIQUE

SYSTEM

```
You identify weaknesses in a draft answer relative to the question.  Return a
brief critique only - no rewrite.

Rules:
1. Return plain text only.
2. List specific gaps (missing facts, wrong claims, incomplete coverage).
3. Maximum 5 bullet points.
```

USER

```
Question:  {query}

Draft answer:  {draft_answer}

Passages:  {retrieved_passages}

List the gaps in the draft answer.
```

REFINE

SYSTEM

```
You rewrite a draft answer to address a critique, using new passages.

Rules:
1. Return plain text only.
2. Correct all issues raised in the critique.
3. Preserve correct content from the draft.
4. Maximum {max_words} words.
```

USER

```
Question: {query}

Draft answer: {draft_answer}

Critique: {critique}

New passages: {retrieved_passages}

Write the refined answer.
```

K.14 DRAFT_ANSWER

Purpose. Phase 0 draft answer prompt. The draft is used (i) to anchor entity verification and (ii) as context for the AMBIGUITY_CRITIQUE call that initialises $\Phi(T_0)$.

SYSTEM

```
You produce an initial draft answer for a complex question.  The draft will be
critiqued and improved - completeness matters more than perfect precision at
this stage.

Rules:
1. Return plain text only.
2. Be comprehensive:  cover all aspects you can identify from the passages.
3. Flag uncertainty with phrases such as "it is unclear whether".
4. Maximum {max_words} words.
5. Do not invent facts; ground every claim in the provided passages.
```

USER

```
Question: {query}

Retrieved passages:
{retrieved_passages}

Write a comprehensive draft answer.
```

K.15 The `fmt()` Helper and Output Discipline

Implementation Note: `fmt()`, Missing Keys, and JSON-Only Output

`fmt()` helper. Every template string is processed by a single `fmt(template, **kwargs)` function before being sent to the model. It performs a single-pass `str.format_map()` using a `SafeMapping` subclass: missing keys are replaced with `<MISSING:key>` rather than raising `KeyError`. This ensures a partially filled prompt is sent (with the missing slot visibly labelled) rather than crashing the pipeline mid-round. The helper logs a warning for every missing substitution so that ablation runners can detect incomplete configurations.

JSON-only output discipline. All prompts that request structured output include an explicit instruction in the **SYSTEM** turn: *"Return ONLY valid JSON — no prose, no markdown fences."* At runtime, the parser strips any leading/trailing whitespace and attempts `json.loads()`. On failure it applies a single repair pass (stripping ‘‘‘`json...`‘‘‘ fences and truncating at the last `}`) and retries. If both attempts fail, the affected round is treated as a rollback and the previous answer is retained.

Entity pinning. The verified entity set `{verified_entities}` is computed once in Phase 0 via a regex NER filter and injected into every subsequent template. Aggregation and rewrite prompts additionally include a hard prohibition on paraphrasing entities. This pin eliminates the entity-drift failure mode reported in prior iterative RAG work.

Length caps. `{max_words}` defaults to 300 for root-level aggregation and 150 for rewrite variants, enforced by post-processing truncation at the nearest sentence boundary.

L Robustness of the coverage residual

The coverage residual depends on both the residual encoder and the granularity of answer segmentation. We examine these two choices separately. The encoder used to measure coverage is conceptually distinct from the dense retriever: changing the former changes the residual geometry without necessarily changing the retrieved corpus or the retrieval index.

L.1 Changing the residual encoder

We replace the default residual encoder, e5-base-v2, with gte-base or a 384-dimensional MiniLM-L6 encoder. For each replacement, we recompute all encoder-dependent quantities: answer coverage, corpus-grounding scores, candidate-facet deduplication, and dead-leaf pruning. The dense retriever, corpus, generation backbone, and prompts are held fixed. Similarity thresholds are recalibrated by percentile matching on unlabeled similarity scores.

Table 16: Sensitivity to the residual encoder on ASQA. Alternative encoders use percentile-matched thresholds. PruneRAG is a reference from the main comparison, not a method rerun under each residual encoder. Rollback is the fraction of attempted rewrites rejected by the coverage gate.

Residual encoder	Dim.	DR	Disambig-F1	Rollback (%)
e5-base-v2 (default)	768	**38.2**	**52.2**	24.7
gte-base	768	37.1	50.5	27.2
MiniLM-L6	384	34.8	46.3	33.3
PruneRAG (reference)	–	33.9	41.4	–

The comparable-dimensional alternative reduces DR by 1.1 points, while the smaller encoder reduces it by 3.4 points. Both configurations remain above the PruneRAG reference in this comparison. Thus, the measured advantage is not restricted to the default encoder, although the residual geometry materially affects its magnitude. These experiments vary the residual encoder; they do not establish invariance to changing the retrieval index or corpus.

The rollback rate increases from 24.7% to 27.2% and 33.3%, respectively. This shows that the alternative geometries reject more candidate rewrites under their own coverage criteria. It does not, by itself, establish that the rejected rewrites are factually worse, because the acceptance signal is still the encoder-dependent proxy.

Threshold transfer. Keeping the original numerical thresholds instead of percentile matching reduces DR by a further 1.3 points for gte-base and 2.9 points for MiniLM-L6. Absolute similarity thresholds therefore should not be assumed transferable across encoder families. Percentile matching is label-free in the sense that it uses similarity distributions without gold answers or evaluation scores. It need not preserve residual magnitudes, aggregate-loss ordering, or downstream accuracy.

L.2 Answer segmentation

Our reference coverage computation uses individual answer sentences as the segments p_j in the coverage map. Each leaf description is an aspect-level sentence. We compare this choice with two-sentence windows and paragraph-level segments by recomputing the coverage map. Agreement is the reported proportion of per-leaf decisions matching the sentence-level reference.

Table 17: Agreement of per-leaf coverage decisions with sentence segmentation. Agreement is a consistency diagnostic, not accuracy against human coverage labels or an end-to-end answer-quality measurement.

Answer segmentation	Agreement (%)
Sentence (reference)	100.0
Two-sentence window	93.4
Paragraph	79.6

Coarser segmentation changes a substantial fraction of coverage decisions, especially at paragraph level. We consequently retain sentence segmentation. This diagnostic does not identify which disagreements are false positives or false negatives; that distinction requires independently annotated coverage.

M Additional backbone and benchmark evaluations

The following experiments extend the primary two-backbone evaluation. Their 500-question subsets are separate evaluation settings; their absolute scores should not be pooled with the main evaluation or interpreted as additional independent runs of it.

M.1 A larger backbone: Qwen3-32B

We evaluate FAIR-RAG, PruneRAG, and GBT-RAG with Qwen3-32B on a 500-question ASQA subset. Table 18 gives the aggregate scores, and Table 19 stratifies Disambig-F1 by the number of gold interpretations.

Table 18: ASQA results with Qwen3-32B on the additional 500-question subset. All scores are percentages; bold indicates the best value among these methods.

Method	DR	ROUGE-L	Disambig-F1
FAIR-RAG	36.8	29.6	45.7
PruneRAG	40.8	**31.7**	52.5
GBT-RAG	**42.5**	31.0	**58.3**

GBT-RAG improves DR by 1.7 points and Disambig-F1 by 5.8 points over PruneRAG, while ROUGE-L is 0.7 points lower. The DR margin is smaller than the 4.3-point margin in the main Llama-3.1-8B comparison. Because the backbone and evaluation subset both differ between these comparisons, the difference in margins is not a controlled estimate of the effect of model scale alone.

Table 19: Disambig-F1 by gold interpretation count with Qwen3-32B on the same 500-question ASQA subset. Gaps are percentage-point differences.

Gold interpretations	PruneRAG	GBT-RAG	Gap
1	59.7	**61.0**	+1.3
2–3	52.7	**59.1**	+6.4
4+	42.6	**53.3**	+10.7

Within this evaluation, the advantage increases with the number of required interpretations. This is consistent with the coverage-demand pattern in the main stratified analysis. It supports a benefit on questions with more interpretations under a larger backbone, rather than a claim that stronger backbones cannot reduce the benefit of structured residuals.

M.2 2WikiMultiHopQA and the IRCoT baseline

We additionally evaluate on 2WikiMultiHopQA [Ho et al., 2020], including IRCoT [Trivedi et al., 2023]. The comparison uses Llama-3.1-8B, a common retriever and corpus, greedy decoding with a 256-token generation cap, and a 500-question subset sampled with seed 42.

GBT-RAG exceeds PruneRAG by 8.4 F1 and 11.5 EM points. IRCoT exceeds the listed sequential and iterative references on this benchmark. This extension covers the five methods and one backbone shown; it does not identify the causal mechanisms of their errors.

Table 20: Additional evaluation on a 500-question 2WikiMultiHopQA subset with Llama-3.1-8B. This table reports the new benchmark separately from the original three-benchmark evaluation.

Method	F1	EM
Standard RAG	32.6	25.9
FAIR-RAG	43.3	34.7
IRCoT	47.4	35.1
PruneRAG	58.8	47.0
GBT-RAG	**67.2**	**58.5**

N Operating budgets and adaptive facet expansion

N.1 Compressed-answer quality and profiled cost

The main compute analysis reports 47.2 seconds per question for the default $M = 3$ configuration on ASQA with Llama-3.1-8B, a single A100-80GB GPU, and batch size one. Table 21 decomposes this latency.

Table 21: Stage-level latency for the default configuration under the ASQA timing setup. Percentages are computed from the reported 47.2-second total; rounded percentages need not sum exactly to 100.

Stage	Latency (s)	Share (%)
Phases 0 and 1: coordinates and initial learner	23.4	49.6
Boosting round 1	9.8	20.8
Boosting round 2	8.1	17.2
Boosting round 3	5.9	12.5
Total	47.2	100.0

Table 22: Post-compression answer scores and cumulative retrievals from Table 3, alongside costs from the recorded $M = 3$ profile. The round-2 time estimate omits the final 5.9-second stage. The profile does not itemize final-compression overhead, so the time estimates do not establish end-to-end costs for separately compressed outputs. Token counts for the truncated trace were not reported.

Trace position	DR	Disambig-F1	Retrievals	Profile (s)
After update $m = 2$	35.8	46.5	24.5	41.3
After update $m = 3$ (full)	38.2	52.2	25.0	47.2

After final compression, the round-2 answer is 2.4 DR points below the round-3 answer. Relative to the initial compressed answer's 27.5 DR, the first two updates retain

$$\frac{35.8 - 27.5}{38.2 - 27.5} \approx 77.6\% \tag{33}$$

of the post-compression gain from $m = 0$ to $m = 3$, using the same initial reference as the Single Round ablation in Table 2. Cumulative retrieval counts at $m = 2$ and $m = 3$ remain 24.5 and 25.0, a 2.0% reduction. Omitting the last stage of the recorded latency profile gives $47.2 - 5.9 = 41.3$ seconds, or $5.9/47.2 = 12.5\%$ less profiled time. This is a truncated-profile estimate, not a separate end-to-end timing of the compressed outputs. Final-compression overhead is not itemized in the supplied profile.

What the budget controls establish. The existing compute analysis includes extended-budget FAIR-RAG (32.4 DR at 31,840 input tokens) and three-restart PruneRAG with gold-informed oracle selection (35.1 DR at 52,620 input tokens), compared with 38.2 DR at 29,180 input tokens for GBT-RAG. These are useful controls at comparable or greater input-token spend. They do not constitute an exactly matched retrieval-call experiment, and matching input tokens does not simultaneously match latency, output tokens, and corpus access. Conclusions about efficiency are restricted to the measured configurations.

N.2 Yield of adaptive coordinate expansion

We summarize the logged outcomes of candidate-facet filtering on ASQA. Table 23 summarizes the reported filtering outcomes. The final two rows are conditional on admission.

Table 23: Adaptive facet-expansion diagnostics. Denominators distinguish proposal-filtering yield from the later utility of admitted leaves.

Outcome	Denominator	Rate (%)
Rejected by deduplication	All proposed facets	34.3
Rejected by corpus grounding	All proposed facets	25.6
Admitted	All proposed facets	40.1
Positive residual at admission	Admitted leaves	88.6
Evidence survives an accepted rewrite	Admitted leaves	72.9

Most admitted leaves have positive residual at admission, and 72.9% are reported to contribute evidence retained in an accepted rewrite. These diagnostics characterize the selectivity and downstream use of the expansion mechanism. They do not measure semantic coverage against gold interpretations, the factual correctness of retained evidence, or the false-rejection rate of the filters. A zero-residual admitted leaf triggers no residual-directed retrieval at that step; this does not imply that it incurs no auditing, embedding, grounding, or bookkeeping cost.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction precisely state the three contributions of GBT-RAG: (1) identifying the iterative/structural integration gap, (2) proposing the GBT-RAG framework with text gradient, residual trees, and monotone gating, and (3) demonstrating consistent empirical gains. All claims are substantiated by Theorem 5.1 and the experiments in Section 6.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 explicitly discusses two open limitations: dependence on a fixed encoder geometry for the coverage residual, and the restriction to depth-two decomposition trees; directions for addressing both are outlined.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Theorem 5.1 states assumptions (A1) and (A2) explicitly and provides a complete proof inline. Theorem A.1 (finite termination and the empty-active-set characterization) is stated and proved in Appendix A.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 6.1 specifies the primary backbones (Llama-3.1-8B-Instruct, Qwen2.5-7B-Instruct), retriever (`e5-base-v2`), corpus (Dec-2018 DPR Wikipedia), evaluation framework (FlashRAG), and benchmarks. Default hyperparameters and task-dependent decomposition prompts are described in Appendix C; sampling and aggregation are specified in Appendix D, and baseline configurations in Appendix E.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is provided in the Supplementary Material. All datasets used (HotpotQA, MuSiQue, ASQA) are publicly available; data access instructions and preprocessing steps are included in the supplemental code release.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: GBT-RAG involves no gradient-based training; all design choices are inference-time hyperparameters ($\tau, \beta, M, k, \gamma^*, \kappa_{\min}, K_{\max}$) reported in Appendix C. Dataset splits follow standard benchmark protocols described in Appendix D.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Table 1 reports means over five evaluation-question samples with greedy decoding. The present paper reports point estimates without error bars or a statistical-test table; we therefore do not claim statistical significance for every metric or both backbones.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Appendix I reports the ASQA/Llama-3.1-8B profile on a single A100-80GB GPU, including per-question latency and token costs. Complete resource profiles for all supplementary backbone and benchmark configurations are not reported.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics?

   Answer: [Yes]

   Justification: This work involves only publicly available NLP benchmarks and open-source language models; no human subjects, sensitive data, or dual-use risks are present. The authors have reviewed and comply with the NeurIPS Code of Ethics.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 7 notes that improved RAG coverage can benefit knowledge-intensive applications (e.g., medical or legal question answering) while also noting that higher-fidelity retrieval could amplify misinformation if the underlying corpus contains biased or false content.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse?

    Answer: [N/A]

    Justification: GBT-RAG is a retrieval and reasoning framework built on top of existing open-source LLMs and public corpora; it does not release new pre-trained models or scraped datasets, and poses no elevated misuse risk beyond the underlying models themselves.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models) used in the paper properly credited, and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets (HotpotQA, MuSiQue, ASQA), models (Llama-3.1-8B-Instruct, Qwen2.5-7B-Instruct, `e5-base-v2`), and the FlashRAG framework are cited in Sections 6.1 and the references.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The GBT-RAG codebase released in the Supplementary Material includes a README with environment setup, installation instructions, dataset preparation steps, commands to reproduce all main-table results, and an MIT license.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects; all evaluations are conducted on pre-existing, automatically annotated benchmarks.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals were obtained?

    Answer: [N/A]

    Justification: No human subjects are involved in this research; IRB approval is therefore not applicable.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [Yes]

    Justification: LLMs ($\mathcal{M}$) are a core component of GBT-RAG, used as the generation oracle for drafting, critique-driven decomposition, aggregation, and evidence-grounded rewriting (Sections 4.2–4.4). The primary backbones are named in Section 6.1, and the additional Qwen3-32B evaluation is described in Appendix M.1.