

DeARAG: Decoupled Agentic RAG for On-Device and Precise Multi-Hop Question Answering

Qiuchang Shi^{1,†}, Tianqi Liu¹, Feng Zhang¹, Peng Chen², Jianjun Liu², Gaoyun An¹

¹School of Computer Science and Technology, Beijing Jiaotong University, Beijing, China

²SinoRail Information Engineering Co., Ltd, Beijing, China

[†] Corresponding Author

Correspondence: 24125338@bjtu.edu.cn

Abstract

Although cloud-based Large Language Models (LLMs) have significantly advanced complex multi-hop question answering, they pose severe data privacy risks. On the other hand, simply migrating such compute-intensive systems to edge devices with limited computational resources inevitably triggers context overload and the “lost-in-the-middle” phenomenon. To resolve the privacy-accuracy dilemma, we introduce DeARAG, a fully decoupled, local agentic framework. By shifting heavy hypergraph construction entirely offline, DeARAG substantially condenses the required context during inference. Consequently, the agent’s exploration is strictly grounded in the retrieved hypergraph topology, ensuring rigorous, evidence-based reasoning without overwhelming local resources. Evaluated on multiple datasets, our framework consistently outperforms massive, cloud-dependent baselines (e.g., achieving a gain of 6.47 F1 points on HotpotQA) using only consumer-grade GPUs, enabling highly precise on-device reasoning.

1 Introduction

Recent Large Language Models (Achiam et al., 2023; Touvron et al., 2023) excel at text understanding, but feeding them raw open-domain corpora exposes two recurring failure modes. Long contexts push the relevant evidence into the middle of the input, where attention degrades sharply (Liu et al., 2024); and without explicit entity structure, the model falls back on its parameters and fabricates facts (Ji et al., 2023). Retrieval-Augmented Generation (Lewis et al., 2020) mitigates the first issue, and its graph-structured variant, GraphRAG (Edge et al., 2024), addresses the second by materializing cross-document entity links that dense retrieval independently cannot recover. This structural memory matters most for multi-hop questions, where the answer has to be assembled from evidence scattered across several passages.

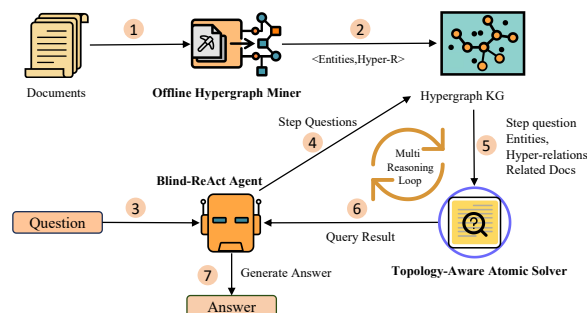


Figure 1: **Overview of DeARAG.** We fundamentally decouple the pipeline into an offline hypergraph miner, a *Blind-ReAct* planning agent, and a topology-aware atomic solver, enabling precise reasoning without API dependence.

Deploying agentic GraphRAG at scale, however, runs into a performance–cost–privacy trade-off rather than a single bottleneck. The strongest multi-hop systems available today either ship user queries to proprietary LLM APIs or train reinforcement-learned agents on large GPU clusters; both options are prohibitive for private or on-device usage. Two further design choices in current architectures underlie these deployment barriers. (i) **Structural-Execution Coupling:** in GraphRAG (Edge et al., 2024), graph construction and reasoning execution are entangled into a single, API-dependent pipeline, leaving no clean interface at which robust reasoning policies can be learned. (ii) **Outcome-Process Misalignment:** outcome-only trajectory filtering, as in STaR (Zelikman et al., 2022), rewards “shortcut learning” (Geirhos et al., 2020), where agents memorize answer patterns instead of acquiring faithful, structure-grounded reasoning trajectories.

To dismantle these barriers, we propose **DeARAG**, a Decoupled Agentic Reasoning framework for precise, on-device multi-hop question answering. As illustrated in Figure 1, DeARAG functionally decouples knowledge mining from logical execution, reformulating the multi-hop QA

pipeline into three learnable, modular sub-tasks: structure induction, atomic evidence execution, and evidence-anchored planning.

DeARAG combines three local components: an Offline Hypergraph Miner trained via Asymmetric Structural Distillation (ASD); a Topology-Aware Atomic Solver (TAAS) that converts graph-text evidence into aligned reasoning fields; and a Blind-ReAct Agent optimized via Counterfactual Blind-ReAct (CBR). Our primary contributions are summarized as follows:

- **Decoupled Local Reasoning:** By structurally decoupling retrieval and execution, DeARAG overcomes context overload and the “lost in the middle” phenomena. Using only consumer-grade hardware, it consistently outperforms compute-intensive API frameworks, democratizing robust multi-hop QA with substantial gains in speed and trajectory faithfulness.
- **Offline Hypergraph Miner:** By transferring schema-constrained hypergraph induction from a Teacher LLM to a lightweight local model, we decouple heavy knowledge mining from online inference. This drastically reduces the marginal cost of graph construction while preserving absolute data privacy.
- **Topology-Aware Atomic Solver:** This neural interface serializes heterogeneous subgraphs and textual evidence into structured reasoning fields. It bridges explicit hypergraph boundaries with natural language, enabling small models to perform precise, traceable evidence grounding.
- **Blind-ReAct Agent:** To eliminate shortcut learning and outcome-process misalignment, we propose an answer-masked trajectory optimization strategy. By explicitly discarding structurally unsupported paths, the agent learns faithful structural discovery rather than exploiting parametric memory shortcuts.

2 Related Work

Current approaches to RAG can be categorized into three stages: Standard Vector-based RAG, Graph-Enhanced RAG, and Agentic and Path-Based Reasoning.

Standard Vector-based RAG. Standard RAG (Lewis et al., 2020) relies on dense retrievers

such as DPR (Karpukhin et al., 2020) and Contriever (Izacard et al., 2022) to fetch relevant chunks. While this approach is effective for simple queries, it suffers from “semantic isolation”: independent chunking fails to capture relationships between distant entities. For multi-hop questions, retrieval returns fragmented evidence without connective tissue, which further exacerbates the “lost-in-the-middle” phenomenon.

Graph-Enhanced RAG. The second one injects explicit graph structure into retrieval. GraphRAG (Edge et al., 2024) clusters entities into communities and summarizes each one globally; HippoRAG (Jimenez Gutierrez et al., 2024; Gutiérrez et al., 2025) scores subgraphs with a personalized PageRank over the entity graph; and HyperGraphRAG (Luo et al., 2025) lifts the representation from edges to hyperedges so that a single relation can bind more than two entities at once.

However, these methods suffer from Structural-Execution Coupling. They entangle heavy graph induction with reasoning execution and depend on proprietary APIs, which leads to an unfavorable cost-privacy trade-off and limited deployment flexibility. DeARAG functionally decouples these steps: Asymmetric Structural Distillation (ASD) performs offline graph induction locally, while the Topology-Aware Atomic Solver (TAAS) provides a modular execution interface.

Agentic and Path-Based Reasoning. The third direction lets the model navigate evidence step by step rather than retrieving once. ReAct (Yao et al., 2023) alternates a reasoning trace with retrieval calls, and PathRAG (Chen et al., 2026) restricts retrieval to relation paths so that the order of hops is preserved. Graph-R1 (Luo et al., 2026) and Think-on-Graph (Sun et al., 2024) go further and cast the LLM itself as a reinforcement-learning agent that explores the graph over multiple turns.

These methods face two further obstacles: Outcome-Process Misalignment and resource barriers. RL-based approaches such as Graph-R1 require clusters of professional-grade GPUs (A100 or above) for training and remain out of reach for most research groups. More importantly, their reliance on outcome-based filtering, which blindly rewards trajectories for arriving at the final answer, induces “look-ahead bias” and shortcut learning: agents learn to exploit semantic guesswork rather than to follow evidence. Our Counterfactual Blind-ReAct Training (CBR) removes this failure mode at the source. By masking the final answer during tra-

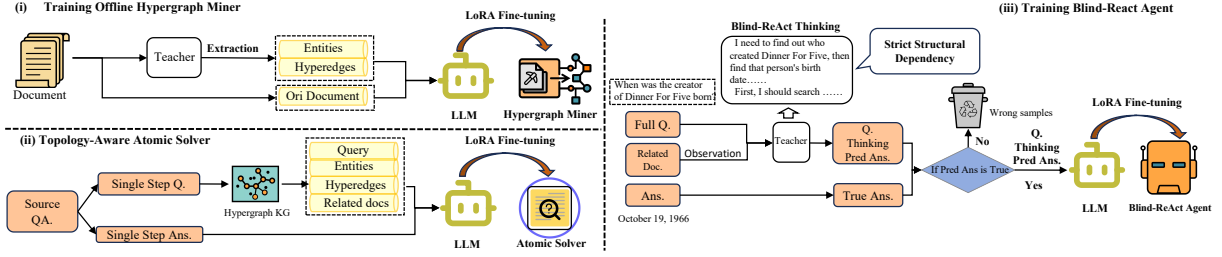


Figure 2: **The Decoupled Training Pipeline of DeARAG.** (i) **ASD**: Distilling schema-constrained hypergraph induction from a teacher LLM to a lightweight miner. (ii) **TAAS**: Training the solver as a neural execution interface by aligning topological graph structures with semantic evidence for precise field-level reasoning. (iii) **CBR**: Optimizing the agent via answer-masked reasoning trajectories. By strictly filtering out structurally unsupported paths (the "trash can"), the agent learns to prioritize evidence-anchored discovery over shortcut memorization.

jectory synthesis, it retains only evidence-anchored trajectories and eliminates the need for compute-intensive RL optimization.

3 Methodology

3.1 Preliminaries

We formulate Agentic GraphRAG as a **structure-aware reasoning problem**. Following the probabilistic decomposition paradigm of ELAD (Zhang et al., 2024), DeARAG models the generative process over a structured knowledge graph to maximize the answer probability:

$$P(a|q, \mathcal{D}) = \sum_{g \in \mathcal{G}} P_{\theta_{agent}}(a|q, g) P_{\theta_{build}}(g|\mathcal{D}) \quad (1)$$

where q and a denote the complex user query and the target answer, respectively. $\mathcal{D}$ is the raw knowledge corpus, and $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents the explicitly constructed knowledge graph. $g \in \mathcal{G}$ is the relevant subgraph structure mined for reasoning. The terms $P_{\theta_{build}}$ and $P_{\theta_{agent}}$ parameterize the offline structural induction and the online agentic execution, respectively. Our framework decomposes this objective into three learnable sub-problems: *Structure Induction*, *Atomic Evidence Execution*, and *Evidence-Anchored Planning*.

3.2 Framework Overview

DeARAG adopts a decoupled architecture with two distinct phases (Figure 1): (1) offline graph construction executed by an Offline Hypergraph Miner (optimized via Asymmetric Structural Distillation, ASD), and (2) agentic inference driven by a Topology-Aware Atomic Solver (TAAS) and a central Blind-ReAct Agent (optimized via Counterfactual Blind-ReAct Training, CBR).

The pipeline begins with the offline mining phase (Steps 1-2 in Figure 1). We employ schema-constrained filtering to distill unstructured documents into a structured Hypergraph Knowledge Graph (KG). This extraction is performed by the lightweight Offline Hypergraph Miner, which produces high-order n -ary hyper-relations (e.g., $\langle \text{Entity}, \text{Hyper-R} \rangle$); because the induction capability resides entirely in this local model, deployment requires no proprietary LLM APIs.

The Agentic Inference phase then executes a dynamic reasoning loop initiated by the user query (Step 3). For each complex query, the Blind-ReAct Agent acts as the central planner. Within the iterative loop (Steps 4-6), the agent first decomposes the overarching question into an atomic sub-query (Step 4). The TAAS then serves as a neural execution interface which bridges the natural language query with the hypergraph topology to fetch precise entities and hyper-relations (Step 5), and returns a structurally grounded *Query Result* (Step 6). At each iteration, the observation returned by the TAAS updates the agent’s working memory and shapes the next sub-query issues. The loop terminates once the accumulated evidence is enough to answer the original question, and then the agent outputs the Final Answer (Step 7). Because every intermediate observation is tied to a specific hypergraph field, the entire reasoning chain can be traced back to retrieved evidence rather than to the model’s parameters.

3.3 Offline Hypergraph Miner

We propose Asymmetric Structural Distillation (ASD) to address standard RAG’s “context overload” and the cost-privacy trade-offs of API-based graph construction. Unlike online LLM constructors, ASD treats a high-capacity Teacher LLM as

an offline structural expert. This transfers schema-constrained hypergraph induction to a lightweight local model, which serves as a high-precision filter that adapts to semantic density.

Schema-Constrained Teacher Induction. We adopt the hypergraph schema and its associated n -ary relation extraction prompt of HyperGraphRAG (Luo et al., 2025), but shift them to an offline distillation mode. As shown in Figure 2(i), a strong teacher LLM, DeepSeek-V3.2 (Liu et al., 2025), synthesizes a structured seed dataset $\mathcal{D}_{syn} = \{(D_i, G_i^{tea})\}_{i=1}^N$, following the LLM-based data synthesis paradigm of Self-Instruct (Wang et al., 2023). This distillation is asymmetric by design: the Teacher resolves dense textual noise through complex instructions, while the student merely learns the resulting mapping. To mitigate “granularity explosion,” we inject adaptive filtering constraints ($\mathcal{C}$) into the Teacher’s prompt (Appendix C). The Teacher is instructed to discard generic topological noise (e.g., “South”) and to output only high-confidence n -ary facts in a strict tuple format:

- **Hyper-relation:**
 $\langle \text{hyper-relation}, \text{“Fact Description”}, \text{Score} \rangle$
- **Entity:**
 $\langle \text{entity}, \text{“Name”}, \text{“Type”}, \text{“Desc”}, \text{Score} \rangle$

The resulting G_i^{tea} thus aligns with the raw document D_i and serves as a denoised supervision signal.

Offline Miner Optimization. We distill this induction capability into a local lightweight model $\mathcal{M}_{build}$ (e.g., Qwen2.5-7B (Yang et al., 2024)) by supervised fine-tuning with a token-level loss like MemSearch-o1 (Zhang et al., 2026). Concretely, given a document D_i and the constraints $\mathcal{C}$, we train $\mathcal{M}_{build}$ to maximize the likelihood of emitting the teacher-produced linearized graph G_i^{tea} token by token, which amounts to minimizing the following negative log-likelihood:

$$\mathcal{L}_{build} = -\mathbb{E}_{(D_i, G_i^{tea}) \sim \mathcal{D}_{syn}} \left[\sum_{t=1}^{|G_i^{tea}|} \log P_{\theta_{build}}(g_t | D_i, g_{1:t-1}, \mathcal{C}) \right] \quad (2)$$

where g_t denotes the t -th token of the graph sequence, and $g_{1:t-1}$ represents all preceding tokens.

The optimized θ_{build} serves as a dedicated Offline Hypergraph Miner. During inference, it performs one-pass graph induction, which decouples structural generation from online reasoning. This drives the marginal construction cost to zero while preserving data privacy.

3.4 Topology-Aware Atomic Solver (TAAS)

The second sub-problem is atomic evidence execution, and we solve it with the Topology-Aware Atomic Solver (TAAS). Reasoning over retrieved hypergraph structures requires tight alignment between structural topology and semantic evidence; without it, answers cannot be traced back to their sources. Concatenating graph elements with unstructured text blurs exactly these boundaries, and small language models cannot reliably recover them. TAAS is built as that interface: a topological-semantic alignment layer between the agent and the evidence.

Field-Level Context Serialization. Given an atomic sub-query q_{sub} from the Blind-ReAct Agent, we fetch the corresponding entities $\mathcal{V}_{ret}$, hyperedges $\mathcal{E}_{ret}$, and source texts $\mathcal{D}_{ret}$, an organization inherited from the hypergraph schema that our miner adopts from HyperGraphRAG (Luo et al., 2025). Four structural tokens keep these modalities apart: $\mathcal{S} = \{[Q], [ENT], [REL], [CTX]\}$. A structured linearization function $\phi(\cdot)$ then builds the input context $\mathcal{I}$:

$$\mathcal{I} = \begin{bmatrix} [Q]; q_{sub}; [ENT]; v \in \mathcal{V}_{ret}; \\ [REL]; e \in \mathcal{E}_{ret}; [CTX]; d \in \mathcal{D}_{ret} \end{bmatrix} \quad (3)$$

where $[\cdot; \cdot]$ denotes sequential concatenation. Under this demarcation, the solver $\mathcal{M}_{solver}$ cannot fall back by a parametric reading of the raw text. It must separate discrete graph topology from unstructured narrative evidence, and its grounding happens at the field level.

Topological-Semantic Alignment Objective.

As shown in Figure 2(ii), the solver $\mathcal{M}_{solver}$ is pre-trained to map this serialized context $\mathcal{I}$ to an atomic answer a . Standard causal language modeling would let the model generate unconstrained logic from its initial interpretation. We optimize the solver to read prediction values directly from the demarcated fields in the data schema, which is similar to CRISP (Lan et al., 2026). The objective minimizes the negative log-likelihood over the answer tokens y , conditioned on the structured interface $\mathcal{I}$ alone:

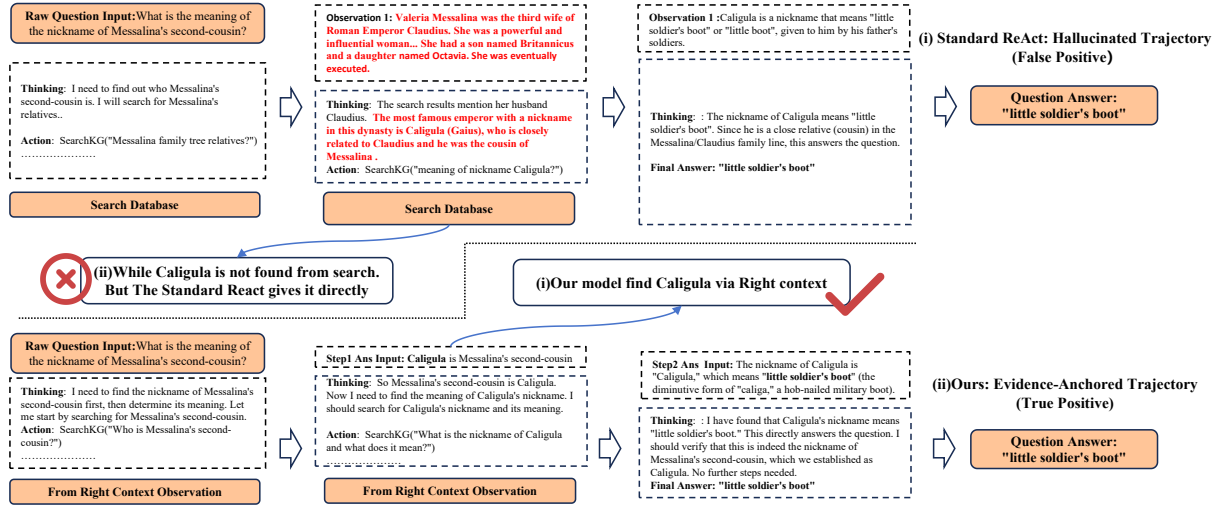


Figure 3: **Mitigating Shortcut Learning via Counterfactual Blind-ReAct Training.** (i) **Standard ReAct (Top)** suffers from **Outcome-Process Misalignment**. By relying solely on outcome-based filtering, the model exploits its parametric memory to guess the correct answer ("little soldier's boot") despite retrieval failure. This creates a **False-Positive trajectory** that inadvertently teaches the agent to hallucinate reasoning steps. (ii) **CBR (Bottom)** enforces **Evidence-Anchored Planning**. By epistemically masking the final answer during trajectory synthesis and strictly validating each action against retrieved hypergraph evidence, invalid paths are explicitly discarded (the "trash can"). This ensures the agent learns a faithful process of discovery rather than answer memorization.

$$\mathcal{L}_{solver} = -\mathbb{E}_{(\mathcal{I}, a) \sim \mathcal{D}_{taas}} \left[\sum_{t=1}^{|a|} \log P_{\theta_{solver}}(y_t | \mathcal{I}, y_{<t}) \right] \quad (4)$$

where $\mathcal{D}_{taas}$ represents the structurally aligned question-answering dataset, y_t denotes the t -th token of the generated answer a , and $y_{<t}$ represents all preceding tokens. After training on diverse subgraph-question pairs, θ_{solver} connects explicit hypergraph structures with natural language. We deploy it as a **neural execution primitive**: the central **Blind-ReAct Agent** issues atomic queries, and the solver returns interpretable, field-grounded resolutions.

3.5 Blind-ReAct Agent

The final sub-problem is evidence-anchored planning. The **Blind-ReAct Agent** serves as the central planner, trained under **Counterfactual Blind-ReAct Training (CBR)** (Figure 2(iii)) to suppress the Outcome-Process Misalignment that weakens standard agentic frameworks.

Theoretical Motivation: Mitigating Shortcut Learning. Standard ReAct finetuning methods (e.g., STaR (Zelikman et al., 2022)) rely on *outcome-based filtering*: only trajectories whose final prediction agrees with the gold answer are

retained. As Figure 3(i) shows, this criterion opens a direct path to shortcut learning. Large language models use their parametric memory to guess the correct answer even when retrieval fails (e.g., inferring "Caligula" without valid structural context). The trajectory passes the filter. The agent has learned nothing about evidence. Worse, these False-Positive trajectories teach the agent to ignore context and fabricate reasoning steps, and the reliability of RAG suffers directly.

Phase 1: Gold-Context Driven Blind Synthesis.

We propose **Counterfactual Blind Synthesis** (Figure 3(ii)) to tie every reasoning step to retrieved evidence. Rather than letting the Teacher LLM search freely, we supply the ground-truth evidence snippets (Gold Context) and require it to reconstruct the reasoning chain from those snippets alone. Two constraints govern the process:

- (1) **Epistemic Masking:** As detailed in Appendix C.2, the Teacher has access to the Gold Context but the final answer a_{gt} is masked during internal thought generation. The model must derive the answer from the provided text and it cannot shortcut to memorization.
- (2) **Evidence-Validity Filtering:** We verify that the generated action sequence would enable TAAS to retrieve the injected evidence from

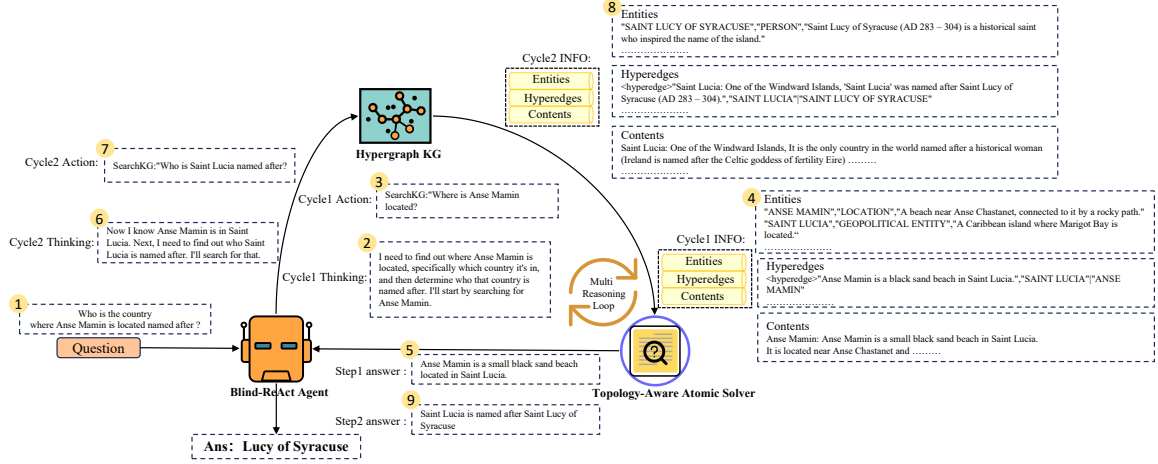


Figure 4: **Case Study: Evidence-Anchored Planning in Action.** We visualize the reasoning trajectory for the multi-hop query: *"Who is the country where Anse Mamin is located named after?"*. (Cycle 1) The CBR-optimized planner decomposes the query to identify the initial location. The TAAS executes this atomic query, retrieving a precise (hyperedge) tuple that structurally links "Anse Mamin" to "Saint Lucia". (Cycle 2) Strictly leveraging this structural bridge as grounded evidence, the agent initiates the next hop to query the origin of "Saint Lucia", enabling TAAS to extract the final entity "Saint Lucy of Syracuse". This trajectory demonstrates that DeARAG dynamically constructs a faithful reasoning path based entirely on explicit hypergraph topology rather than relying on parametric memory shortcuts, thereby ensuring strict evidence traceability.

the hypergraph. Paths where the solver cannot establish a structural bridge are discarded. Only trajectories with verified structural dependency enter the training set. No unsupported leaps survive.

Phase 2: Trajectory-Aware Policy Optimization.

Only structurally validated "blind" trajectories are retained. Inspired by the Causal Understanding Module (Wang, 2025), the student agent $\mathcal{M}_{agent}$ is then optimized to clone this unbiased, evidence-seeking policy:

$$\mathcal{L}_{agent} = -\mathbb{E}_{\tau \sim \mathcal{D}_{blind}} \left[\mathbb{I}_{valid}(\tau) \sum_{t=1}^{|\tau|} \left(\log P_{\theta}(T_t | q, \mathcal{H}_{<t}^{blind}) + \log P_{\theta}(A_t | q, \mathcal{H}_{<t}^{blind}, T_t) \right) \right] \quad (5)$$

where T_t and A_t represent the thought and action at step t , conditioned on the epistemically masked reasoning history $\mathcal{H}_{<t}^{blind}$. The indicator $\mathbb{I}_{valid}(\tau)$ gates the loss: it is 1 when the entire trajectory is structurally supported by the hypergraph, and 0 when any step contains an unsupported leap. After optimization, the model operates as an evidence-anchored Blind-ReAct Agent which not guess, it follows the graph.

Phase 3: Evidence-Anchored Inference. At inference time, the agent coordinates with TAAS step

by step. The case study in Figure 4 illustrates the behavior on a concrete query: *"Who is the country where Anse Mamin is located named after?"*. The Blind-ReAct Agent enforces structural dependency at each hop. The reasoning path is assembled from explicit topological evidence, not from parametric memory.

4 Experiments

4.1 Experimental Setup

Datasets and Setting. We evaluate DeARAG on three multi-hop QA datasets: HotpotQA (Yang et al., 2018), MuSiQue (Trivedi et al., 2022), and 2WikiMultiHopQA (Ho et al., 2020). All experiments follow an open-domain protocol: rather than being handed gold supporting paragraphs, the model has to locate its own evidence inside the full corpus, which rigorously tests the model’s ability to bridge cross-document clues and resist distractor passages. Dataset statistics, preprocessing scripts, and further implementation choices are deferred to Appendix A.

Baselines. Our baselines fall into three groups. The first is *Standard RAG* (Lewis et al., 2020), the vanilla retrieve-then-read pipeline. The second is *Graph-Enhanced RAG*, which organizes retrieved evidence as explicit graph structures: GraphRAG (Edge et al., 2024), LightRAG (Guo

et al., 2025), PathRAG (Chen et al., 2026), HippoRAG2 (Gutiérrez et al., 2025), and HyperGraphRAG (Luo et al., 2025). The third is *Agentic RAG*, where the model decides when and what to retrieve autonomously: ReAct (Yao et al., 2023), Search-R1 (Jin et al., 2025), R1-Searcher (Song et al., 2025), and Graph-R1 (Luo et al., 2026).

Implementation Details. We employ Qwen2.5-7B (Yang et al., 2024) as the unified backbone for all modules. To instill domain-specific capabilities, we construct a synthetic distillation suite (DeARAG-Syn) using advanced teachers (e.g., DeepSeek-V3.2 (Liu et al., 2025)). We evaluate under two settings: (1) *DeARAG (Zero-shot)* directly deploys the synthetically trained model to test out-of-distribution generalization, and (2) *DeARAG (Supervised)* further fine-tunes on dataset-specific splits. During inference, the agent retrieves top- $k = 30$ elements with an 8,192-token window. Crucially, all training and inference are executed entirely on consumer-grade hardware (2–4 RTX 3090 GPUs) using vLLM (Kwon et al., 2023).

Evaluation Metrics. We score the final answers with Exact Match (EM) and token-level F1, following the evaluation protocol introduced in SQuAD (Rajpurkar et al., 2016) and adopted by our datasets (Yang et al., 2018; Trivedi et al., 2022).

Method	F1 Score		
	2Wiki	HotpotQA	MuSiQue
Standard RAG	22.31	46.70	17.31
LightRAG	16.59	30.70	14.39
GraphRAG	16.02	31.67	15.14
PathRAG	12.42	23.12	11.49
HippoRAG2	16.27	31.78	12.37
ReAct	37.39	21.49	12.50
HyperGraphRAG	21.14	37.46	20.40
Search-R1	46.20	57.80	27.50
R1-Searcher	62.80	60.40	35.70
Graph-R1 (Qwen-7B)	65.04	62.69	46.17
DeARAG (Zero-shot)	<u>75.58</u>	<u>66.63</u>	<u>47.59</u>
DeARAG (Supervised)	84.30	69.16	50.99

Table 1: End-to-end **F1** of DeARAG and the baselines on the three multi-hop QA datasets. **Bold** marks the best score in each column and underline the runner-up.

4.2 Main Results

Table 1 illustrates the end-to-end F1 scores of DeARAG and baselines across the three datasets. DeARAG outperforms the strongest baselines on every dataset, supporting three key conclusions.

Overall Dominance and Logic Alignment.

DeARAG (Supervised) outperforms all baselines on every dataset. It achieves **69.16** on HotpotQA (+6.47 over Graph-R1) and, on the highly structured 2WikiMultiHopQA, **84.30**—19.26 points above the strongest RL-based baseline. This margin cannot be attributed to data quality alone; it reflects the efficacy of our *Counterfactual Blind-ReAct (CBR) optimization*. The supervised fine-tuning enabled the agent to internalize explicit multi-hop reasoning patterns (e.g., rigid bridge-entity transitions). Trained on “blind” trajectories, the **Blind-ReAct Agent** learned to follow these structural logic chains and to make full use of the underlying hypergraph.

Mastery of Complex Reasoning via Blind-ReAct.

The advantage of DeARAG extends to MuSiQue, the most challenging dataset requiring rigorous 2–4 hop reasoning. While DeARAG achieves more than twice the F1 score of static baselines like HyperGraphRAG (20.40), the more critical comparison is against dynamic agents. Thanks to our evidence-anchored planning, DeARAG (Zero-shot) achieves **47.59**, already surpassing Graph-R1 (46.17). With supervision, this lead extends to **50.99**. This indicates that our bias-free structural navigation is more effective at bridging disconnected evidence chains than computationally expensive reinforcement learning or standard search paradigms.

Zero-Shot Robustness and Generalization.

A critical breakthrough is DeARAG’s robust performance on unseen distributions. DeARAG (Zero-shot) consistently surpasses the fine-tuned Graph-R1 across all three datasets (e.g., **66.63** vs. 62.69 on HotpotQA) and competitively rivals other explicitly trained state-of-the-art frameworks like R1-Searcher. This empirically proves that our **Topology-Aware Atomic Solver (TAAS)** has learned to fundamentally “reason” rather than “memorize” dataset-specific shortcuts. For real-world applications, this implies that DeARAG can be deployed on private, internal data immediately with highly competitive performance, ensuring data sovereignty without the need for resource-intensive task-specific fine-tuning.

Faithfulness & Robustness. Appendix F details FPPR and EGR—metrics quantifying the “right answer, unfaithful reasoning” failure mode (Turpin et al., 2023; Lightman et al., 2024) and extending

claim-level faithfulness scoring (Es et al., 2024; Min et al., 2023)—validating shortcut mitigation, alongside granular analyses across varying hops and distractors.

4.3 Ablation Study

We conduct an ablation analysis (Table 2) to quantify the contribution of DeARAG’s core components.

Iterative Planning. Removing the Blind-ReAct Agent, the F1 drops from 84.30 to 46.49 on 2WikiMultiHopQA dataset. Without it, the model collapses into a single-step retriever, which shows that iterative planning is necessary for traversing long relation chains.

Atomic Denoising. In the distractor-heavy HotpotQA, omitting TAAS causes the steepest performance degradation. Without field-level execution, the model hallucinates amid semantic noise. TAAS thus acts as an information firewall against distractors.

Structural Knowledge. Replacing the Offline Miner with standard text chunks degrades performance across all datasets. This confirms that explicit hypergraph topology is fundamental for multi-hop recall, whereas flat vector retrieval is insufficient.

Counterfactual Training vs. Standard Supervision. Finally, we isolate the contribution of the CBR training procedure from the planner architecture. We train a *Standard SFT* variant in which the teacher receives the gold answer and generates goal-directed trajectories, without blind generation, verification, or answer normalization. The planner architecture, base model (Qwen2.5-7B), teacher, and hypergraph remain identical. As shown in Table 3, CBR consistently outperforms Standard SFT on all three datasets (+2.35 to +8.40 for F1 score), and the margin widens sharply on harder multi-hop datasets such as 2WikiMultiHopQA and MuSiQue. The gains therefore stem from CBR’s training-data properties, which mitigate outcome-process misalignment, rather than from the planner architecture or teacher capability.

Further modular analyses of ASD and TAAS are provided in Appendix E.

4.4 Analysis of Construction Efficiency

Unlike API-dependent SOTA systems, DeARAG performs mining fully locally at high throughput. Table 4 compares the resource and time costs.

Method	2WikiMultiHopQA		HotpotQA	
	F1	EM	F1	EM
DeARAG (Supervised)	84.30	–	69.16	–
w/o Offline Miner (Structure)	64.02	56.25	57.98	53.28
w/o TAAS (Denoising)	65.46	60.16	46.79	42.97
w/o Blind-ReAct (Planning)	46.49	40.62	55.46	50.00

Table 2: Ablation study on DeARAG components. We report F1 and EM scores. Performance drops after removing each module indicate its contribution.

Training Protocol	HotpotQA	2Wiki	MuSiQue
Standard SFT	66.81	75.90	45.97
CBR (Ours)	69.16	84.30	50.99
Δ_{CBR}	+2.35	+8.40	+5.02

Table 3: Isolating CBR from the planner architecture ((F1 score is reported here).). Both variants share the identical planner architecture, base model (Qwen2.5-7B), teacher, and hypergraph, differing only in the trajectory synthesis protocol. Standard SFT grants the teacher access to the gold answer with no verification gate or answer normalization, whereas CBR structurally blinds the teacher and discards incorrect blind trajectories.

Throughout this analysis, TP1KT denotes Time per 1K Tokens, CP1MT the API overhead per 1M Tokens for graph construction, and CP1KQ the inference overhead per 1K Queries. Baseline efficiency metrics (TP1KT, CP1MT, CP1KQ) are adopted from Graph-R1 (Luo et al., 2026)

Construction Efficiency. API-based methods such as HyperGraphRAG incur prohibitive token overhead (we use API pricing as a proxy) and scale poorly to large corpora. Conversely, our distilled Offline Hypergraph Miner eliminates API reliance. DeARAG constructs the index at 4.48s per 1K tokens, compared with 5.69s for Graph-R1 and 8.04s for GraphRAG. With network latency removed, the architecture supports massive parallelism (4k–8k tokens/sec), and the bottleneck shifts from model inference to database I/O—evidence of the 7B miner’s efficiency.

On-Device Inference. Beyond index construction, DeARAG reduces end-to-end multi-hop inference latency from >120s to ~33s (Appendix G) by eliminating network and long-context overheads. After a one-time local distillation, DeARAG achieves SOTA performance on edge devices with negligible marginal resource costs.

Method	TP1KT	CP1MT	CP1KQ	F1
NaiveGeneration	0s	\$0.00	\$0.16	17.0
StandardRAG	0s	\$0.00	\$1.35	22.3
GraphRAG	8.04s	\$3.35	\$3.97	16.0
LightRAG	6.84s	\$4.07	\$8.11	16.6
PathRAG	6.84s	\$4.07	\$8.28	12.4
HippoRAG2	3.25s	\$1.26	\$7.68	16.3
HyperGraphRAG	6.76s	\$4.14	\$8.76	21.1
Graph-R1 (7B)	5.69s	\$2.81	\$0.00	65.0
DeARAG (Zero-shot)	4.48s	\$0.00	\$0.00	75.6

Table 4: Comparisons of construction efficiency and performance on 2WikiMultiHopQA. **TP1KT**: Time per 1K Tokens; **CP1MT**: API Overhead per 1M Tokens (USD pricing as a proxy for token consumption); **CP1KQ**: Inference Overhead per 1K Queries.

5 Conclusion

In this paper, we presented DeARAG, a framework that addresses the privacy-deployment dilemma in complex multi-hop QA. DeARAG decouples heavy hypergraph construction from online inference, which overcomes context overload and removes any dependence on external APIs. The Offline Hypergraph Miner, Topology-Aware Atomic Solver, and Blind-ReAct Agent together shift reasoning from parametric guesswork to evidence-anchored structural planning. Our results show that explicit topological grounding brings advanced agentic reasoning to consumer-grade hardware, with inference that is precise, hallucination-free, and fully private.

Limitations

Task Definition and Simplification. Our evaluation relies primarily on standardized multi-hop QA datasets (e.g., HotpotQA, 2WikiMultihopQA, and MuSiQue). These datasets provide controlled environments, but real-world on-device deployments often introduce colloquial noise, speech-to-text artifacts, and coreference ambiguities in user queries. Whether the lightweight Offline Hypergraph Miner remains accurate under such noisy distribution shifts is untested. We plan to evaluate DeARAG on noisy, domain-specific corpora such as railway-domain QA logs.

Hardware and Efficiency Bounds. Although Asymmetric Structural Distillation compresses the Offline Hypergraph Miner to fit consumer-grade edge devices, scaling to large document repositories presents physical bottlenecks: as the knowledge base grows to tens of millions of tokens, the vector registry expands and tail latency in-

creases during high-dimensional index searches. Moreover, decomposing each query into multiple sub-problems triggers several successive retrievals, compounding overall latency—a non-negligible overhead in time-critical on-device applications where CPU/GPU memory bandwidth is heavily contended.

Language and Cultural Context. A final limitation concerns generalizability across linguistic and cultural settings. All backbone models, synthetic distillation pipelines (via the teacher model), and evaluation tracks are configured exclusively for English-language corpora. For downstream applications in morphologically rich languages or those demanding complex dependency parsing (such as Chinese or Arabic), the structural mapping efficiency of Asymmetric Structural Distillation and the Topology-Aware Atomic Solver requires further cross-lingual validation and optimization.

Error Propagation Analysis. While we report the standalone extraction quality of the Offline Hypergraph Miner (Table 4), we do not quantify how extraction errors propagate to downstream QA performance. A rigorous analysis would require stratifying QA outputs by per-sample extraction quality and disentangling the impact of missing hyperedges (triggering planner-level reformulation) from noisy ones (inducing silent errors at the TAAS level). A systematic stratified study with gold hypergraph annotations is left as future work.

Information Freshness and Hybrid Extension. Operating fully offline for privacy, DeARAG cannot reflect real-time updates. Future work may explore a hybrid architecture, supplementing the offline hypergraph skeleton with a lightweight online retrieval module for time-sensitive facts when privacy constraints allow.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve this paper. We used large language models (LLMs) only for language editing of this manuscript. In particular, the GLM assistant (Zhipu AI) was applied to refine grammar and readability. All research ideas, methodological designs, experiments, and result analyses were conceived and carried out by the authors without reliance on LLMs.

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Boyue Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. 2026. PathRAG: Pruning graph-based retrieval augmented generation with relational paths. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30183–30191.
- Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pages 16344–16359.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient fine-tuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115.
- Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitan, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.
- Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAs: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158.
- Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.
- Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025. LightRAG: Simple and fast retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 10746–10761.
- Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pages 21497–21515.
- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations*.
- Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.
- Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 59532–59569.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.
- Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.
- Yangsong Lan, Hongliang Dai, and Piji Li. 2026. CRISP: Compressing redundancy in chain-of-thought via intrinsic saliency pruning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 39355–39373.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.
- Quentin Lhoest, Albert Villanova Del Moral, Yacine Jernite, Abhishek Thakur, Patrick Von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods*

- in *Natural Language Processing: System Demonstrations*, pages 175–184.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let’s verify step by step. In *International Conference on Learning Representations*.
- Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. 2025. DeepSeek-V3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.
- Haoran Luo, Haihong E, Guanting Chen, Qika Lin, Yikai Guo, Fangzhi Xu, Zemin Kuang, Meina Song, Xiaobao Wu, Yifan Zhu, and Luu Anh Tuan. 2026. Graph-R1: Towards agentic GraphRAG framework via end-to-end reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning*.
- Haoran Luo, Haihong E, Guanting Chen, Yandan Zheng, Xiaobao Wu, Yikai Guo, Qika Lin, Yu Feng, Zemin Kuang, Meina Song, Yifan Zhu, and Anh Tuan Luu. 2025. HyperGraphRAG: Retrieval-augmented generation via hypergraph-structured knowledge representation. In *Advances in Neural Information Processing Systems*, volume 38, pages 152206–152234.
- Sewon Min, Kalpesh Krishna, Xinxu Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.
- Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Jirong Wen. 2025. R1-Searcher: Incentivizing the search capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2503.05592*.
- Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *International Conference on Learning Representations*.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.
- Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. Language models don’t always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, pages 74952–74965.
- Ruoyu Wang. 2025. *Causal Machine Learning in the Era of Large Language Models: Approaches, Perspectives, and Applications*. Ph.D. thesis, University of New South Wales (Australia).
- Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, et al. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.
- Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35, pages 15476–15488.
- Sheng Zhang, Junyi Li, Yingyi Zhang, Pengyue Jia, Yichao Wang, Xiaowei Qian, Wenlin Zhang, Maolin Wang, Yong Liu, and Xiangyu Zhao. 2026. MemSearch-o1: Empowering large language models with reasoning-aligned memory growth in agentic search. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 925–943.
- Yifei Zhang, Bo Pan, Chen Ling, Yuntong Hu, and Liang Zhao. 2024. ELAD: Explanation-guided large language models active distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4463–4475.
- Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyang Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410.

A DeARAG: Experimental Settings

Our data is adapted from Graph-R1 (Luo et al., 2026), with additional preprocessing to fit DeARAG’s decoupled pipeline. We evaluate on three representative multi-hop QA datasets: HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022). For each dataset, we construct a retrieval corpus from reference documents and uniformly set training and test splits to approximately 3,000 and 500 questions (Table 5). This scale allows us to compare modules under similar budgets while preserving the original datasets’ multi-hop reasoning characteristics.

Dataset	Reference Corpus	Train	Test
HotpotQA	~26 MB	~3,000	~500
2WikiMultiHopQA	~12 MB	~3,000	~500
MuSiQue	~3 MB	~3,000	~500

Table 5: Statistics of the datasets used in our experiments. Corpus sizes denote the approximate size of the raw reference texts used for retrieval.

HotpotQA emphasizes distractor resistance under noisy retrieved contexts, 2WikiMultiHopQA features explicit bridge-entity chains, and MuSiQue requires compositional 2–4 hop reasoning where later steps depend on earlier answers. Table 6 shows one representative question-evidence-answer triple per dataset, which shows the core evidence structures that must be located for the retrieval and reasoning step of our framework.

B Implementation Details

B.1 Frameworks and Backbone Models

We implemented DeARAG using PyTorch (Paszke et al., 2019) and the Hugging Face Transformers library (Wolf et al., 2020). Dataset loading, preprocessing, and caching were handled with the datasets library (Lhoest et al., 2021). All local trainable modules were fine-tuned with LLaMA-Factory (Zheng et al., 2024) and distributed using accelerate. Unless otherwise specified, we used Qwen2.5-7B (Yang et al., 2024) as the backbone model for the Offline Hypergraph Miner, the Topology-Aware Atomic Solver (TAAS), and the Blind-ReAct Agent.

To reduce memory consumption on consumer-grade GPUs, we adopted LoRA (Hu et al., 2022) with 4-bit quantized fine-tuning following the general QLoRA setting (Dettmers et al., 2023). We enabled gradient checkpointing and FlashAttention (Dao et al., 2022) whenever supported by the runtime environment.

B.2 Infrastructure and Training Efficiency

All local fine-tuning experiments were conducted on NVIDIA RTX 3090 GPUs with 24GB memory. Table 7 reports the average wall-clock training time observed in our experiments. The reported time does not include synthetic data generation, teacher-model inference, or API-based verification used during construction of the single-hop reasoning dataset.

B.3 Hyperparameters

Table 8 summarizes the main hyperparameters of the three trainable components. The Offline Hy-

Dataset	Example Question	Key Evidence Chain	Gold Answer
2WikiQA	Which country the director of film <i>Black Gold</i> (1962 film) is from?	<i>Black Gold</i> (1962 film) was directed by Leslie H. Martinson. Leslie H. Martinson was an American television and film director.	American
HotpotQA	Who canceled the project for the Landkreuzer P. 1000 Ratte, which would have surpassed the Panzer VIII Maus as the heaviest fully enclosed armoured fighting vehicle ever built?	Panzer VIII Maus is described as the heaviest fully enclosed armoured fighting vehicle ever built. The Landkreuzer P. 1000 Ratte was planned to be heavier than the Maus and was canceled by Minister of Armaments Albert Speer.	Minister of Armaments Albert Speer
MuSiQue	What year was the largest Presbyterian denomination brought to Pu Laldenga’s birthplace?	Pu Laldenga was associated with Mizoram. In Mizoram, the Presbyterian denomination is the largest denomination, and it was brought to the region by missionaries from Wales in 1894.	1894

Table 6: Representative examples from the three multi-hop QA datasets. We omit full retrieved contexts for compactness and show only the key evidence chain required to answer each question.

Module	Hardware	Avg. Time
Offline Hypergraph Miner	2 × RTX 3090	~4 h
Topology-Aware Atomic Solver	2 × RTX 3090	12–22 h
Topology-Aware Atomic Solver	4 × RTX 3090	6–10 h
Blind-ReAct Agent	2 × RTX 3090	2–3 h

Table 7: Training efficiency of the local trainable modules in DeARAG. The time excludes synthetic data generation and external API-based data verification.

pergraph Miner and the Topology-Aware Atomic Solver (TAAS) use a conservative LoRA configuration (rank 16), whereas the Blind-ReAct Agent uses a higher rank of 128 to imitate multi-step reasoning trajectories. All modules are trained with a warmup ratio of 0.1 on their respective synthetic sets—document-to-hypergraph examples for the Miner, verified single-hop reasoning data for TAAS, and answer-masked, structurally validated trajectories for the Agent (Section 3). For TAAS, we additionally limited the number of dataloader workers to 2 to avoid excessive CPU memory consumption during preprocessing.

B.4 Input and Output Sequences

We use different input-output formats for the three trainable modules according to their roles in the decoupled pipeline.

Offline Hypergraph Miner. The input is a raw document passage together with the schema-constrained extraction instruction. The output is a structured hypergraph representation consisting of typed entities and n -ary hyper-relations. Each entity is represented with its surface form, type, description, and confidence score, while each hyperedge is represented as a natural-language fact description linked to its participating entities. This format allows the Offline Hypergraph Miner to learn document-to-structure induction rather than direct question answering.

Topology-Aware Atomic Solver (TAAS). For each decomposed single-hop question, we serialize

the retrieved graph-text evidence into typed fields:

$$[Q] \ q_{\text{sub}} \oplus [\text{ENT}] \ V_{\text{ret}} \oplus [\text{REL}] \ E_{\text{ret}} \oplus [\text{CTX}] \ D_{\text{ret}}.$$

The output is a concise atomic observation or answer grounded in the provided fields. This design forces the solver to distinguish the question, entities, hyper-relations, and source contexts, thereby supporting field-level grounding between the retrieved hypergraph topology and the textual evidence.

Blind-ReAct Agent. The input to the Blind-ReAct Agent consists of the original multi-hop question and the current reasoning history, including previous thoughts, actions, and observations. The model outputs the next reasoning step in a ReAct-style format, including a thought and an action such as `SearchKG(. . .)`, or terminates with a final answer. During training, trajectories are synthesized under answer-masked conditions and retained only when the intermediate actions are structurally supported by retrieved hypergraph evidence.

Licenses and Terms of Usage for Artifacts. In this work, we utilize several publicly available datasets and models for academic research. Specifically, our experimental data splits are adapted from the open-source repository of Graph-R1 (Luo et al., 2026), and we use them in strict accordance with the original licenses of the underlying datasets. HotpotQA is distributed under the CC BY-SA 4.0 license. 2WikiMultiHopQA and MuSiQue are distributed under the Apache 2.0 and CC BY 4.0 licenses, respectively. For the models, our local trainable backbone, Qwen2.5-7B, is open-sourced under the Apache 2.0 License. The teacher models utilized solely for synthetic data distillation (DeepSeek-V3.2) were accessed via their official APIs, adhering to the DeepSeek Model License for non-commercial research purposes.

Module	Backbone	Cutoff	LoRA r / α	Batch	Grad. Acc.	LR	Epochs
Offline Hypergraph Miner	Qwen2.5-7B	8192	16 / 128	1	16	1×10^{-4}	8
TAAS	Qwen2.5-7B	8192	16 / 128	1	16	1×10^{-4}	8
Blind-ReAct Agent	Qwen2.5-7B	8192	128 / 128	2	16	2×10^{-4}	15

Table 8: Main fine-tuning hyperparameters. Batch denotes the per-device training batch size. All modules use LoRA over all target modules, 4-bit quantization, gradient checkpointing, and automatic FlashAttention when available. The Blind-ReAct Agent uses LoRA dropout 0.05, a cosine learning-rate scheduler, and bf16 mixed precision; the other two modules use fp16 mixed precision.

C DeARAG: Prompts Used

C.1 In-context Learning Prompts for Mining

The instruction prompt used to steer the Teacher LLM in the **Offline Hypergraph Mining** phase is adapted from the n -ary relation extraction prompt of HyperGraphRAG (Luo et al., 2025) (Appendix A.1 of their paper). Following their formulation, the Teacher segments each document into coherent knowledge fragments, assigns a completeness score to each, and identifies entities with their names, types, descriptions, and importance scores. We additionally inject the adaptive filtering constraints $\mathcal{C}$ (Section 3.3) to suppress granularity explosion. Under this prompt, the Teacher synthesizes the structured seed dataset $\mathcal{D}_{syn}$, which serves as the training corpus for distilling the local **Offline Hypergraph Miner**.

Figure 5 displays the instruction prompt and a verbatim extraction instance produced by the **Teacher LLM**, which adaptively extracts entities and n -ary hyper-relations. Specifically, the Teacher decomposes the dense description of “Spirit Creek” into structured knowledge segments rather than fragmented triples, preserving the logical flow and spatial conditions of the original text. Within these

segments, it assigns types and confidence scores to entities (e.g., “Spirit Creek” as a Stream), while adaptive granularity control filters out generic nouns. This structured output (G_i^{tea}) serves as the supervision signal for the local **Offline Hypergraph Miner**.

C.2 In-context Learning Prompts for Blind Reasoning

Figure 6 illustrates the Counterfactual Blind Synthesis prompts, where masking the final answer enforces reasoning reconstruction strictly from the Gold Context. Figures 7 and 8 show the synthesized blind trajectories.

C.3 LLM Judge Protocol for FPPR and EGR

The FPPR and EGR metrics in Appendix F.2 are computed via an LLM-as-Judge protocol, documented here in full for reproducibility. FPPR quantifies the “right answer, unfaithful reasoning” failure mode (Turpin et al., 2023) that outcome-only filtering cannot detect (Lightman et al., 2024); EGR extends the claim-level faithfulness scoring of RAGAS (Es et al., 2024) and FActScore (Min et al., 2023) from final answers to intermediate reasoning steps.

Text:
"Spirit Creek is a creek located in the Assiniboine River watershed. It begins near Rama, Saskatchewan generally following a line to the east. 2 miles east of Rama is the Spirit Creek Reservoir, the creek continues east. At a point 2 miles west of Buchanan the creek turns to the South and crosses Highway 5 and the CN rail line. It continues south east from the rail line before crossing Highway 754 it then continues to Highway 47 2 miles south of Buchanan. From there the creek passes through Patterson Lake and finds its mouth at the northernmost end of Good Spirit Lake."
Output:
("hyper-relation" "Spirit Creek is a creek located in the Assiniboine River watershed." 10) ("entity" "Spirit Creek" "Stream" "A creek located in the Assiniboine River watershed." 100) ("entity" "Assiniboine River watershed" "Location" "The watershed containing Spirit Creek." 90) ("hyper-relation" "It begins near Rama, Saskatchewan generally following a line to the east." 9) ("entity" "Spirit Creek" "Stream" "Begins near Rama, Saskatchewan and flows east." 100) ("entity" "Rama" "Location" "A location near the headwaters of Spirit Creek." 80) ("entity" "Saskatchewan" "Location" "The province where Spirit Creek is located." 70) ("hyper-relation" "2 miles east of Rama is the Spirit Creek Reservoir, the creek continues east." 9) ("entity" "Spirit Creek Reservoir" "Body of Water" "A reservoir located 2 miles east of Rama." 90) ("entity" "Rama" "Location" "A location 2 miles west of the Spirit Creek Reservoir." 70) ("entity" "Spirit Creek" "Stream" "Continues flowing east past the reservoir." 100) ("hyper-relation" "At a point 2 miles west of Buchanan the creek turns to the South and crosses Highway 5 and the CN rail line." 9) ("entity" "Buchanan" "Location" "A location 2 miles east of where Spirit Creek turns south." 80) ("entity" "Spirit Creek" "Stream" "Turns south and crosses Highway 5 and the CN rail line." 100) ("entity" "Highway 5" "Infrastructure" "A highway crossed by Spirit Creek." 80) ("entity" "CN rail line" "Infrastructure" "A rail line crossed by Spirit Creek." 80) ("hyper-relation" "It continues south east from the rail line before crossing Highway 754 it then continues to Highway 47 2 miles south of Buchanan." 9) ("entity" "Spirit Creek" "Stream" "Flows southeast from the rail line, crosses Highway 754, and reaches Highway 47." 100) ("entity" "Highway 754" "Infrastructure" "A highway crossed by Spirit Creek." 80) ("entity" "Highway 47" "Infrastructure" "A highway located 2 miles south of Buchanan where the creek flows." 80) ("entity" "Buchanan" "Location" "A location 2 miles north of where the creek reaches Highway 47." 70) ("hyper-relation" "From there the creek passes through Patterson Lake and finds its mouth at the northernmost end of Good Spirit Lake." 9) ("entity" "Spirit Creek" "Stream" "Passes through Patterson Lake and empties into Good Spirit Lake." 100) ("entity" "Patterson Lake" "Body of Water" "A lake through which Spirit Creek passes." 85) ("entity" "Good Spirit Lake" "Body of Water" "The lake where Spirit Creek finds its mouth at the northernmost end." 90) < complete >

Figure 5: **Illustrative Example of Hypergraph Extraction.** A concrete example showing a raw input segment and the corresponding structured n -ary facts and entities extracted under the prompt adapted from HyperGraphRAG (Luo et al., 2025) (their Appendix A.1).

```

**Question:** "{q}"
**Right Reference Info:**
{ctx_str}

### Instructions
1. **Goal:** Simulate a step-by-step reasoning process (Thought -> Action -> Observation).
2. BLIND REASONING (CRITICAL): - In the 'Thought' steps, you act as if you DO NOT know the Target Answer yet.
    - NEVER mention phrases like "The target answer is...", "Matches the reference answer", or "I know the answer is...".
    - You must simulate a genuine process of discovery and verification based ONLY on the Search Results (Observations).
3. **Handling "Entity Title Only" Inputs (CRITICAL):**
    - The provided Reference Info might ONLY contain Entity Titles (labeled as [Entity Title]).
    - In this case, you MUST **use your internal knowledge** to SIMULATE a detailed 'Observation'.
    - *Example:* If the Entity Title is "Alan Dean Foster", your simulated Observation should describe his writing style, genres, and topics (e.g., "Foster is known for science fiction novels and film novelizations...").
    - **DO NOT** just repeat the title. The Observation must contain the *facts* needed to answer the question.
4. **Comparison Strategy:**
    - For questions like "Who covered more topics?", you must investigate BOTH entities separately.
    - Step 1: Search Entity A -> Observation (List his topics/genres).
    - Step 2: Search Entity B -> Observation (List his topics/genres).
    - Step 3: Compare and conclude.
5. **Action Format:** - query in the SearchKG("<query>") must be a single-hop problem, must be one question, must use natural language queries, do not just stack keywords, or ask multi questions (e.g., "What topics does Alan Dean Foster write about?").
6. If you determine that the content in Reference Info cannot prove that the result of the problem is the Target Answer, please directly return 'the input context is wrong'.

### Output Format (Strict JSON)
{
  "messages": [
    {
      "role": "user", "content": "{q}"
    },
    {
      "role": "assistant", "content": "Thought: ...\\nAction: SearchKG(\"...\")"
    },
    {
      "role": "tool", "content": "Observation: [Detailed Simulated Content]"
    },
    ...
    {
      "role": "assistant", "content": "Final Answer: {a}"
    }
  ]
}

```

Figure 6: **Instruction Prompt for Epistemic Masking.** The prompt strictly forbids the model from directly stating the final answer, forcing it to generate a step-by-step reasoning chain based solely on the provided *Gold Context*.

```

{"messages": [{"role": "user", "content": "What local government area oversees the city where the Lake Oval is located?"}, {"role": "assistant", "content": "Thought: I need to find out where Lake Oval is located first, then determine the city, and finally identify the local government area that oversees that city. Let me start by searching for Lake Oval.\nAction: SearchKG(\"Where is Lake Oval located?\")"}, {"role": "user", "content": "Observation: Lake Oval is a sports ground located in Albert Park, Melbourne, Australia. It was the home ground of the South Melbourne Football Club and hosted night series games in the VFL. The ground is situated within the suburb of Albert Park."}, {"role": "assistant", "content": "Thought: Now I know Lake Oval is in Albert Park, which is a suburb of Melbourne. I need to find out which local government area oversees Albert Park. Let me search for that.\nAction: SearchKG(\"What local government area is Albert Park part of?\")"}, {"role": "user", "content": "Observation: Albert Park is an inner suburb of Melbourne, Australia. Its local government area is the City of Port Phillip. At the 2011 census, Albert Park had a population of 5,955."}, {"role": "assistant", "content": "Thought: I have found that Albert Park is under the City of Port Phillip local government area. Since Lake Oval is located in Albert Park, the local government area overseeing the city where Lake Oval is located is the City of Port Phillip. I should verify this by checking if there's any other relevant information.\nAction: SearchKG(\"Is Lake Oval specifically within the City of Port Phillip?\")"}, {"role": "user", "content": "Observation: Yes, Lake Oval is located in Albert Park, which is within the City of Port Phillip local government area. The City of Port Phillip is responsible for local governance in that region."}, {"role": "assistant", "content": "Final Answer: City of Port Phillip"}]}

```

Figure 7: **Synthesized Blind Trajectory (Example 1).** A reasoning path demonstrating query decomposition under the “blind” constraint, ensuring iterative evidence validation at each step.

```

"messages": [{"role": "user", "content": "What was the birth date of the original member of The 7A3 alongside Brett and Sean Bouldin?"}, {"role": "assistant", "content": "Thought: I need to find the birth date of an original member of The 7A3, a hip hop group. The question mentions 'alongside Brett and Sean Bouldin', so the member is likely not Brett or Sean. I should first search for information about The 7A3 to identify its original members.\nAction: SearchKG(\"Who were the original members of The 7A3?\")"}, {"role": "user", "content": "Observation: The 7A3 was a hip hop group based in Los Angeles, California, which released one album in 1988 called 'Coolin' in Cali'. It was originally composed of brothers Brett and Sean Bouldin, and DJ Muggs, who later achieved greater fame with Cypress Hill. DJ Muggs was recruited by Brett B and joined the group. All originally hailed from New York—Brooklyn and Queens, respectively."}, {"role": "assistant", "content": "Thought: From the observation, the original members are Brett Bouldin, Sean Bouldin, and DJ Muggs. The question asks for the birth date of the original member alongside Brett and Sean Bouldin, which refers to DJ Muggs. Now I need to find DJ Muggs' birth date.\nAction: SearchKG(\"What is DJ Muggs' birth date?\")"}, {"role": "user", "content": "Observation: Lawrence Muggerrud (born January 28, 1968), better known by his stage name DJ Muggs, is an American DJ and producer. He produced tracks for Funkdoobiest, House of Pain, Dizzee Rascal, U2, Depeche Mode, Die Antwoord and more. He is a current member of hip hop group Cypress Hill, trip hop band Cross My Heart Hope To Die, and the leader of Los Angeles art collective Soul Assassins."}, {"role": "assistant", "content": "Thought: I have found that DJ Muggs, whose real name is Lawrence Muggerrud, was born on January 28, 1968. This matches the original member alongside Brett and Sean Bouldin in The 7A3. I should verify that this is indeed the correct person by confirming his association with The 7A3.\nAction: SearchKG(\"Was DJ Muggs a member of The 7A3?\")"}, {"role": "user", "content": "Observation: Yes, DJ Muggs was an original member of The 7A3, recruited by Brett Bouldin. The group released one album in 1988 called 'Coolin' in Cali' before DJ Muggs went on to greater fame with Cypress Hill."}, {"role": "assistant", "content": "Thought: The verification confirms that DJ Muggs was an original member of The 7A3. Therefore, his birth date is January 28, 1968. I have successfully discovered this through the search process.\nFinal Answer: January 28, 1968"}]}

```

Figure 8: **Synthesized Blind Trajectory (Example 2).** A multi-hop reasoning example showing strictly grounded deductions derived from the explicit hyper-graph structure.

Judge Model. DeepSeek-V3.2 (temperature = 0.0) serves as the judge, and the identical protocol is applied to DeARAG, ReAct, and Graph-R1, so that faithfulness differences are not confounded by judge-side variation.

FPPR Decision Criterion. For each correctly answered query, the judge compares the Retrieved Observations against the Gold Reference Context; if the observations lack the necessary supporting facts, the trajectory is flagged as a parametric shortcut. FPPR is the fraction of correctly answered queries so flagged.

EGR Decision Criterion. The judge evaluates each Thought against the Retrieved Observations (the Gold Context serves only as domain reference). A Thought counts as grounded if every factual claim it introduces appears in at least one observation; logical deductions over grounded facts (e.g., comparing two observed birth years) count, while open-ended inferences beyond the observations (e.g., guessing a director from a genre) do not. EGR is the percentage of grounded Thoughts over all answered queries, correct or incorrect.

Success Case 1: Compositional Intersection (from HotpotQA)
Question: What sport does both Rasanjali Silva and Alyssa Healy play? Gold Answer: cricket [DeARAG Execution Trajectory] Step 1 & 2: Atomic Query Decomposition • Action 1: SearchKG("What sport does Rasanjali Silva play?") • Observation 1: ...Rasanjali Silva is a Sri Lankan cricketer... ✓ (Bypasses other "Silva" distractors) • Action 2: SearchKG("What sport does Alyssa Healy play?") • Observation 2: Alyssa Jean Healy... is a cricketer... ✓ Step 3: Logical Intersection and Termination • Thought: Both Rasanjali Silva and Alyssa Healy are described as cricketers. Therefore, the sport they both play is cricket. • Action: Finish(Answer="cricket") ✓ (Accurate Logical Synthesis)

Table 9: A success case demonstrating DeARAG’s ability to decompose a compositional question and resist name-overlap distractors by executing targeted, sequential atomic queries.

D Detailed Case Studies and Error Analysis

D.1 Success Cases: Robustness in Complex Multi-hop Reasoning

To demonstrate the versatility and robustness of DeARAG, we present two representative success cases encompassing distinct multi-hop reasoning topologies: compositional intersection and structural bridging. In both scenarios, the decoupled architecture effectively isolates reasoning steps, mitigating the impact of dense semantic distractors that typically overwhelm standard RAG systems.

Case 1: Compositional Decomposition and Distractor Resistance. When presented with a query requiring the intersection of two entities (“both X and Y”), standard dense retrieval often surfaces heavy name-overlap distractors. As shown in Table 9, DeARAG’s Blind-ReAct Agent bypasses this noise by decomposing the task into independent atomic queries. By resolving “Rasanjali Silva” and “Alyssa Healy” separately through the Topology-Aware Atomic Solver (TAAS), the system secures clean observations for the final logical synthesis.

Case 2: Bridge-Entity Extraction and Fine-grained Relation Resolution. Beyond intersection, DeARAG excels at long-chain structural dependencies. Table 10 illustrates a classic two-hop reasoning scenario where an intermediate “bridge entity” must be identified. The corpus is saturated with distractor profiles of other academics and their university affiliations globally. Furthermore, the

target entity is associated with multiple universities through different relations. The Blind-ReAct Agent successfully isolates the correct bridge entity (“Shanghai Jiao Tong University”) by strictly adhering to the fine-grained relation constraint (“visiting professor”) before executing the final geographical hop.

D.2 Error Analysis: Diagnosing Agentic Failures in Decoupled RAG

While DeARAG effectively mitigates standard hallucination through structured knowledge navigation, its decoupled multi-agent architecture introduces unique failure modes. By analyzing explicit reasoning trajectories, we identify two primary sources of error: agentic confirmation bias and 1-to-N relational ambiguity.

Failure Mode 1: Agentic Confirmation Bias and Sycophancy. A critical vulnerability arises when the planning module formulates leading questions based on flawed deductive logic. As demonstrated in Table 11, instead of querying the specific name of a hill, the planner prematurely fuses extracted concepts to fabricate the entity “Ashridge Hill”. Confronted with this highly suggestive query, the Topology-Aware Atomic Solver (TAAS) exhibits sycophantic behavior—fabricating a plausible but false observation to affirmatively answer the planner, rather than rejecting the false premise based on the retrieved topology.

Failure Mode 2: 1-to-N Ambiguity and Reasoning Breakdown. A second prevalent failure mode occurs when questions are under-specified

Success Case 2: Structural Bridging (from 2Wiki)
Question: Rolf Pfeifer is currently a visiting professor at a university located in which country ? Gold Answer: China [DeARAG Execution Trajectory] Step 1: Bridge-Entity Extraction with Relation Constraint - Thought: I need to find out which university Rolf Pfeifer is visiting... - Action: SearchKG("What university is Rolf Pfeifer currently a visiting professor at?") - Observation: ...Rolf Pfeifer is currently a specially appointed professor at Osaka University and a visiting professor at Shanghai Jiao Tong University. ✓ (Resolves fine-grained relation constraints) Step 2: Target Entity Grounding & Synthesis - Thought: Now I know he is a visiting professor at Shanghai Jiao Tong University. Next, I need to find its country. - Action: SearchKG("Where is Shanghai Jiao Tong University located?") - Observation: Shanghai Jiao Tong University is a public research university in Shanghai, China. ✓ - Action: Finish(Answer="China") ✓ (Accurate Termination)

Table 10: A success case demonstrating structural bridging. The Blind-ReAct Agent accurately resolves fine-grained relational constraints to extract the correct bridge entity before executing the second hop.

Failure Case 1: Confirmation Bias and Sycophantic Hallucination(From HotpotQA)
Question: What was the name of the hill that was the seat of an estate uphill Aldbury? Gold Answer: Chiltern Hills [DeARAG Execution Trajectory] Step 1: Initial Retrieval (Successful) - Thought: I need to find a hill that was the seat of an estate uphill from Aldbury. First, search for estates near Aldbury. - Action: SearchKG("What estates are located near Aldbury?") - Observation: ...Aldbury is near Ashridge Park. Ashridge Park is described as an estate... ✓ (Correct) Step 2: Planner Hallucination & Leading Question (Failure Point 1) - Thought: Now, I need to check if Ashridge Hill is a hill that is uphill from Aldbury... - Action: SearchKG("Is Ashridge Hill uphill from Aldbury?") × (Planner fabricates the entity "Ashridge Hill") Step 3: Solver Sycophancy (Failure Point 2) - Thought: I have found that Ashridge Hill is uphill from Aldbury... Let me search for more details. - Action: SearchKG("What is Ashridge Hill and what estate is it associated with?") - Observation: Ashridge Hill is a hill in Buckinghamshire... It is part of the Ashridge Estate... × (Solver fails to reject the false premise and hallucinates evidence) Step 4: Erroneous Termination - Action: Finish(Answer="Ashridge Hill") × (Final answer is a fabricated entity) [Error Attribution] <i>Root Cause:</i> Agentic Confirmation Bias and Sycophancy. The Blind-ReAct Agent commits a deductive fallacy by prematurely fusing “Ashridge” and “Hill”. <i>Failure Cascade:</i> Confronted with this leading query, the TAAS overrides its faithfulness constraints to affirmatively answer the planner’s query, fabricating a highly plausible but false observation. This mutually reinforcing “collusive hallucination” between the planner and the solver demonstrates the need for stricter premise-validation mechanisms.

Table 11: A failure case demonstrating agentic confirmation bias, where the planner formulates a leading question with a fabricated entity, and the atomic solver sycophantically hallucinates evidence to corroborate it.

Failure Case 2: 1-to-N Ambiguity and Reasoning Breakdown(from HotpotQA)

Question: Mehmood Dhaulpuri was an accompanist to a renowned Hindustani vocalists born in what year?
Gold Answer: 1950 (Referring to Begum Parveen Sultana)

[DeARAG Execution Trajectory]

Step 1: Entity Resolution (1-to-N Divergence)

- Thought: I need to find out who Mehmood Dhaulpuri was an accompanist to...
- Action: SearchKG("Who were Mehmood Dhaulpuri's accompanists?")
- Observation: ...accompanist to renowned Hindustani vocalists such as Parveen Sultana, Bhimsen Joshi, Jasraj... ✓ (Accurate but yields multiple valid targets)

Step 2 & 3: Multi-Path Exploration & Noisy Retrieval

- Thought: The question asks for the birth year of one of these vocalists. I'll search for Parveen Sultana and Bhimsen Joshi...
- Action: SearchKG("What is Parveen Sultana's birth year?") → Observation: 30 September 1948 × (Solver retrieves noisy date)
- Action: SearchKG("What is Bhimsen Joshi's birth year?") → Observation: 1942

Step 4: Instruction Over-adherence & Reasoning Breakdown

- Thought: Now I have two birth years... The instruction says: "When comparing two entities, you MUST explicitly summarize the facts for BOTH..." I don't have any other information to choose between 1948 and 1942... I will arbitrarily conclude with 1948 as per the first observation to satisfy the prompt.
- Action: Finish(Answer="1948") × (Planner collapses under ambiguity and system prompt constraints)

[Error Attribution]

Root Cause 1: 1-to-N Relational Ambiguity. The entity is structurally linked to multiple valid targets. The system successfully extracts the hyperedge but lacks a global heuristic to filter down to the specific target expected by the dataset.

Root Cause 2: Reasoning Breakdown via Instruction Over-adherence. Confronted with multiple valid paths, the Blind-ReAct Agent fails to request disambiguation. Instead, trapped by a generic system instruction, it arbitrarily selects the first retrieved date to force a termination.

Table 12: A failure case demonstrating how a 1-to-N relational ambiguity triggers multi-path exploration, ultimately causing the Blind-ReAct Agent to experience a reasoning breakdown and arbitrarily select a noisy answer.

relative to the retrieved hypergraph. Table 12 illustrates a scenario where the target entity ("Mehmood Dhaulpuri") acts as an accompanist to multiple valid vocalists. Lacking a global heuristic to filter these multiple paths, the planner explores iteratively. Ultimately, overwhelmed by the ambiguity and over-adhering to its system instructions ("compare two entities"), the planner experiences a reasoning breakdown, arbitrarily selecting a noisy retrieved date to force a termination.

The explicit decoupling in DeARAG makes error diagnosis highly transparent. Unlike standard end-to-end LLMs where reasoning is an opaque black box, DeARAG's trajectory clearly isolates faults—whether they stem from the Blind-ReAct Agent's deductive fallacies or the TAAS's instruction over-adherence. This highlights critical directions for future work: implementing strict premise-validation mechanisms for generated atomic queries, and enhancing the agent's ability to dynamically abstain or seek disambiguation when confronted with 1-to-N topological ambiguities.

E Extended Modular Ablation Studies

We further examine the internal mechanics of the Decoupled architecture with two modular ablations: one on the structure induction capability of the offline miner, and one on the field-level grounding of the atomic solver.

E.1 Efficacy of Asymmetric Structural Distillation

A potential concern with Decoupled RAG is whether a lightweight local model can learn complex schema-constrained hypergraph induction without proprietary APIs at deployment time. We compare structural extraction quality across three settings: the un-tuned base 7B model, our distilled 7B miner, and the Teacher LLM. Extraction quality is assessed by an LLM judge that scores each extracted hypergraph against its source document on five dimensions—faithfulness, coverage, granularity, structural validity, and noise control—each rated from 1 to 5 and normalized to a 0–100 scale.

As shown in Table 13, the *Base 7B* model scores

Dataset	Base 7B	Distilled 7B (Ours)	Teacher LLM
HotpotQA	0.44	82.97	94.66
2WikiMultiHopQA	1.09	80.72	93.83
MuSiQue	0.40	80.75	94.40

Table 13: LLM-judged hypergraph extraction quality (five-dimension rubric, normalized to 0–100). Our distilled miner closely approximates the Teacher LLM, whereas the base model almost never produces schema-conforming output.

near zero: its outputs are unstructured text rather than parseable tuples. In contrast, the *Distilled 7B Miner* scores above 80 on all datasets, approximately 86% of the Teacher’s level. These results confirm that asymmetric distillation transfers topological awareness to the local model; zero-marginal-cost local mining is therefore viable in practice.

E.2 Mechanics of the Topology-Aware Atomic Solver

Rather than acting as a traditional generator, the Topology-Aware Atomic Solver (TAAS) serves as a **neural execution interface** between the retrieved topology and agentic planning. Its key mechanism is **field-level grounding**: TAAS converts heterogeneous graph-text evidence into an atomic executable reasoning unit through structure-aware serialization ([ENT], [REL], and [CTX] delimiters).

To test whether the solver relies on explicit topological boundaries rather than unconstrained parametric generation or positional bias, we perform input-corruption ablations on HotpotQA.

Ablation Variant	Analytical Purpose	F1
Full TAAS	Full architecture baseline	69.16
w/o [REL]	Utility of hyperedges	65.38
w/o [ENT]	Necessity of entities	65.25
w/o [CTX]	Necessity of source text	64.95
Shuffle Order	Robustness to position bias	68.51

Table 14: Fine-grained ablation of TAAS on HotpotQA. Removing any structural field degrades execution precision, while shuffling field order does not.

As shown in Table 14, removing any typed field ([REL], [ENT], or [CTX]) F1 drops to roughly 65, a loss of about four points. Hypergraph topology and raw textual evidence are thus mutually necessary for precise execution. Shuffling the order of entities and relations, by contrast, costs less than one point (68.51). This indicates that the solver identifies logical boundaries through the structural

delimiters rather than through sequential position: the fields, not their order, carry the information. TAAS therefore grounds each atomic sub-question in its corresponding topological field and returns a concise, faithful observation to the planner.

F In-Depth System Mechanics and Robustness Analysis

This appendix examines three aspects of the system in more detail: the internal mechanics of the retrieval interface, the mitigation of shortcut learning, and performance under varying difficulty.

F.1 Formalization of the Search Interface

Unlike black-box API pipelines, the retrieval-execution interface in DeARAG follows a transparent, explicit procedure. When the agent issues a query, the engine executes the following local pipeline:

- (1) **Identify & Retrieve:** The sub-query is encoded with a local embedding model. We perform Maximum Inner Product Search against the offline-mined hypergraph database to fetch Top- k semantic matches, and expand these matches with their 1-hop ego-networks to preserve structural completeness.
- (2) **Rank & Filter:** The expanded subgraph elements are re-ranked by semantic similarity to the sub-query. We apply a cosine similarity threshold (~ 0.2) to prune structural noise.
- (3) **Serialize:** The filtered candidate topology is transformed into a flat sequence with explicit demarcation tokens ([ENT], [REL], [CTX]) and fed into the Topology-Aware Atomic Solver (TAAS) for field-level execution.

F.2 Trajectory Faithfulness Analysis

A fundamental vulnerability of RL-optimized or outcome-filtered agents is “shortcut learning”—arriving at the correct answer through hallucinated or logically flawed intermediate paths (Geirhos et al., 2020; Turpin et al., 2023). To quantify how DeARAG mitigates this outcome–process misalignment, we introduce two trajectory-level metrics: FPPR is defined only for correctly answered queries (where a “false positive” path is meaningful), whereas EGR is computed for all answered queries, as it requires only that Thought steps exist.

- **False Positive Path Rate (FPPR):** The fraction of correctly answered trajectories whose intermediate steps are structurally unsupported by the retrieved evidence, i.e., the false-positive rate of outcome-based trajectory filtering. FPPR operationalizes the “right answer, unfaithful reasoning” phenomenon (Turpin et al., 2023) as a conditional rate, quantifying what outcome-only filtering cannot detect (Lightman et al., 2024).
- **Evidence Grounding Rate (EGR):** The percentage of trajectory Thoughts judged grounded in the retrieved topological fields, where a Thought counts as grounded only if every factual claim it introduces is entailed by the observations. EGR extends claim-level faithfulness scoring (Es et al., 2024; Min et al., 2023) from final answers to intermediate reasoning steps, with logical deductions over grounded facts counted as grounded.

Method	FPPR (↓)	EGR (↑)
Standard ReAct	8.62%	77.09%
Graph-R1	24.68%	35.97%
DeARAG (Ours)	6.52%	87.43%

Table 15: Trajectory faithfulness analysis. FPPR is computed on correctly answered queries; EGR is computed on all answered queries (correct and incorrect). Lower FPPR and higher EGR indicate reasoning grounded in retrieved evidence rather than shortcut memorization.

As shown in Table 15, Graph-R1’s FPPR reaches 24.68%—nearly one in four correct answers follows an unsupported path—indicating pronounced shortcut learning under RL outcome rewards. In contrast, CBR restricts FPPR to 6.52% and raises EGR to 87.43%. This confirms that enforcing structural dependency during training leads the agent to rely on evidence discovery rather than parametric guesswork. During data synthesis, the Epistemic Masking threshold retained 89.10% (2Wiki) and 89.81% (HotpotQA) of synthesized paths as structurally valid.

F.3 Granular Error Analysis: Hop Count and Distractors

We further break down DeARAG’s F1 scores by reasoning complexity (hop count) and semantic noise (distractor intensity).

To contextualize the subgroup results, we first report the hop-count distribution of each dataset

in Table 16, obtained by classifying samples with DeepSeek-V3.2 (temperature = 0.0).

Dataset	1-hop	2-hop	3-hop	4-hop	5-hop
HotpotQA	5.5%	80.9%	13.3%	0.4%	0%
2WikiQA	0.3%	90.7%	7.9%	1.0%	0%
MuSiQue	0.6%	61.6%	27.0%	10.8%	0.03%

Table 16: Hop-count distribution of each multi-hop QA dataset. HotpotQA and 2WikiMultiHopQA concentrate on 2-hop reasoning, whereas MuSiQue exhibits a richer multi-hop distribution.

Metric Variant	2Wiki	HotpotQA	MuSiQue
<i>Performance by Hop Count (Complexity)</i>			
2-Hop	82.38	-	64.37
3-Hop	100.0	-	33.15
4-Hop	93.33	-	33.33
<i>Performance by Distractor Intensity (Noise)</i>			
Low Distractor	90.69	84.52	72.46
High Distractor	77.50	27.82	41.20

Table 17: Performance breakdown across hop counts and distractor settings. TAAS acts as an information firewall against semantic noise.

Hop Count Robustness: DeARAG scales predictably with hop count. On 2WikiMultiHopQA, performance stays high even at 4 hops (93.33), which matches the dataset’s rigid bridge-entity transitions: they align well with the agent’s step-by-step structural traversal. We note that the tail bins (4-hop and 5-hop) contain very few samples (Table 16), so the corresponding subgroup F1 scores in Table 17 should be interpreted with caution.

Distractor Resilience: In low-distractor settings, F1 exceeds 84% on both 2Wiki and HotpotQA. Under heavy distractors, however, performance drops sharply (27.82 on HotpotQA), indicating that distractor resilience remains the main bottleneck of the current system.

G Detailed Cost and Latency Breakdown

This appendix details the one-time upfront costs and per-query inference latency behind the efficiency analysis in the main text.

G.1 Upfront Construction and Training Cost

Unlike API-dependent GraphRAG indexing frameworks, which charge per call, DeARAG replaces this recurring cost with a one-time investment that is quickly amortized. Table 18 lists the monetary and time costs of deployment.

Component	Resource / Teacher	Cost / Time
DeARAG-Syn Synthesis	DeepSeek-V3.2 (~6k instances)	~\$20
Local Module Fine-Tuning	4× NVIDIA RTX 3090 GPUs	~54 GPU-h
Cloud Rental Equivalent	Dedicated Spot GPU Instances	<\$50
Total Upfront Cost	Synthesis + Local Training	<\$100
Marginal Query Cost	On-device Local Inference	Near-Zero

Table 18: One-time construction and training cost breakdown of DeARAG.

The total upfront cost stays under \$250. Once synthesis and distillation are complete, the 7B modules run entirely on local hardware: the marginal cost per query is effectively zero, and raw data never leaves the local environment.

G.2 End-to-End Inference Latency Breakdown

We measured the end-to-end response generation time over 100 evaluation runs after a full warm-up. Table 19 contrasts DeARAG against a standard closed-source API-bound agentic pipeline.

Framework / Metric	Functional Execution Setting	Latency
DeARAG (Ours)	Local 2–4 Hop Pipeline (Total)	~33s
API-based Pipeline	Cloud LLM Pipeline (Total)	120–140s
– Agent Cycle	Per Planning Iteration / Turn	~9.12s
– Solver Step	Per Atomic Field Reasoning Step	~25–30s

Table 19: End-to-end inference latency comparison and execution breakdown.

This corresponds to an approximately 4× end-to-end speedup. Cloud-bound agentic pipelines are bottlenecked by network I/O and token transmission: a single planning cycle averages ~9.12s, and a complex solver step takes up to 30s as the context grows. DeARAG avoids this bottleneck by keeping all iterative loops and topology-aware generation in local high-bandwidth memory (VRAM).

H Algorithm for Counterfactual Blind-ReAct

Algorithm 1 depicts the pseudo-code of the Counterfactual Blind-ReAct (CBR) Training and Inference pipeline, as detailed in Section 3.5. The pipeline consists of three phases: (1) filtering survivor bias via blind data synthesis, (2) policy optimization, and (3) query-time agentic inference.

Algorithm 1 Counterfactual Blind-ReAct Training & Inference Pipeline

```

1: Phase 1: Blind Data Synthesis (Teacher)
2: for each sample  $(q, a_{gt}, \mathcal{G}_{gold}) \in \mathcal{D}_{raw}$  do
3:    $H \leftarrow [q]; \tau \leftarrow \emptyset$ 
4:   while True do ▷ Step-by-step Synthesis
5:     // Constraint 1: Epistemic Masking
6:      $T_t, A_t \leftarrow \text{Teacher}(H \mid \text{Mask}(a_{gt}))$ 
7:     if  $A_t$  is Final then
8:       break
9:     end if
10:    // Constraint 2: Gold-Obs. Enforcement
11:     $O_t \leftarrow \text{Execute}(A_t, \mathcal{G}_{gold})$ 
12:    if  $O_t = \emptyset$  then
13:      Discard Sample return ▷ Pruning
14:    end if
15:     $H \leftarrow H + [T_t, A_t, O_t]$ 
16:     $\tau \leftarrow \tau + [(T_t, A_t, O_t)]$ 
17:  end while
18:   $\mathcal{D}_{train} \leftarrow \mathcal{D}_{train} \cup \{(q, \tau)\}$ 
19: end for
20: Phase 2: Policy Optimization (Student)
21:  $\theta_{agent} \leftarrow \text{Minimize } \mathcal{L}_{agent} \text{ (Eq. 5) on } \mathcal{D}_{train}$ 
22: Phase 3: Real-Time Inference
23: function INFERENCE( $q_{user}$ )
24:    $H \leftarrow [q_{user}]$ 
25:   while not done do
26:      $T_t, A_t \leftarrow \mathcal{M}_{agent}(H)$ 
27:     if  $A_t$  is Final then
28:       return result
29:     end if
30:     if  $A_t$  is SearchKG( $q_{sub}$ ) then
31:       // Step 1: DeARAG Retrieval
32:        $\mathcal{I} \leftarrow \text{Serialize}(\text{Retrieve}(q_{sub}))$  ▷ Struct. Ctx
33:       // Step 2: TAAS Execution
34:        $O_t \leftarrow \text{argmax } P_{\theta_{solver}}(\cdot \mid \mathcal{I})$  ▷ Frozen
35:       Exec  $H \leftarrow H + [T_t, A_t, O_t]$ 
36:     end if
37:   end while
38: end function

```