

PROBE: Prospective Retrieval with Oracle-Based End-Goal Distillation for Multi-Hop Question Answering

Xu Lu^{1,4}, Zhengqing Zang⁴, Binhui Yao^{1,4*}, Luwei Yang³,
Ruimin Chen³, Zeng Wang³, Haobo Wang^{1,2,4}

¹School of Software Technology, Zhejiang University

²Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security

³Zhejiang Geely Holding Group Co., Ltd.

⁴State Key Laboratory of Blockchain and Data Security, Zhejiang University

{xu.lu, zangzq, 12321292, wanghaobo}@zju.edu.cn {Luwei.Yang1, Ruimin.Chen, Zeng.Wang3}@geely.com

Abstract

Multi-hop question answering (MHQA) requires models to retrieve and reason over evidence distributed across multiple documents. However, training dense retrievers for MHQA remains challenging because labeled query-document pairs are difficult to obtain for the evolving intermediate queries in multi-hop reasoning. Recent approaches use LLM likelihoods for step-wise supervision over queries conditioned on accumulated evidence, which anchors retrieval to known facts and biases supervision toward answer-bearing documents, leaving bridging evidence undiscovered. To address this, we propose **PROBE**¹, a framework that optimizes dense retrievers for MHQA without requiring manually annotated query-document pairs. It decouples the reasoning state into a factual summary and a prospective query that anticipates missing information. For training, we transfer cross-encoder signals only at the final hop, providing trajectory-level supervision without penalizing early exploration. Experiments on three MHQA benchmarks show that PROBE consistently improves both retrieval and QA performance over strong iterative baselines, achieving an average gain of 5.7 points while using only about 5% of the training data of the prior method.

1 Introduction

Multi-hop Question Answering (MHQA) requires models to aggregate and reason over evidence distributed across multiple documents in order to answer complex queries (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022). Since the complete evidence chain is rarely retrievable in a single step, recent systems typically adopt iterative Retrieval-Augmented Generation (RAG). In this paradigm, models alternate between retrieval and reasoning

across multiple dependent steps, conditioning each new search on previously gathered evidence to progressively construct a reasoning trajectory (Trivedi et al., 2023; Jeong et al., 2024).

Despite the flexibility of iterative reasoning, the performance of multi-hop QA systems remains fundamentally constrained by the underlying retriever. Although dense retrievers such as Contriever (Izacard et al., 2022) capture richer semantic relations than sparse methods like BM25 (Robertson et al., 1994), training them for multi-hop reasoning remains difficult because relevance labels are not readily available for the evolving intermediate queries encountered during iterative reasoning. To address this, ReSCORE (Lee et al., 2025) proposes a label-free framework that trains a dense retriever using LLM generative likelihoods integrated with iterative reasoning. At each hop, the model extracts verified information into a compact thought appended to the next query. However, without explicit forward-looking intent, queries can remain anchored to accumulated evidence, making missing bridging information difficult to target. Moreover, step-wise generative supervision can favor answer-bearing documents over intermediate bridging evidence, further suppressing exploratory retrieval. Figure 1 illustrates such a failure mode, where retrieval remains trapped around surface-form entities across multiple hops.

To address this limitation, we propose **PROBE**, which combines **P**rospective **R**etrieval with **O**racle-**B**ased **E**nd-Goal Distillation to train dense retrievers for MHQA without relying on manually annotated query-document pairs. Instead of relying solely on accumulated evidence, PROBE explicitly models the information needed for the next hop. Concretely, it separates the intermediate context into two components: a *factual summary* that consolidates verified evidence and a **prospective query** that provides a forward-looking directive toward the next missing piece of informa-

*Corresponding author.

¹Our implementation is available at: <https://github.com/luxu-zju/PROBE>

Question: Who created the NBC sitcom that Johnny Pemberton appears in as the character Bo Thompson?	
ReSCORE	PROBE (Ours)
Hop 1 Retrieval: [... , Johnny Pemberton, Bobb'e J. Thompson, Brian Thompson, Fred Thompson, Thompson (TV series), ...]	Hop 1 Retrieval: [... , Johnny Pemberton, Bosom Buddies, Fred Thompson, Johnny Thompson, Thompson (TV series), ...]
Hop 1 Thought: “The NBC sitcom that Johnny Pemberton appears in as the character Bo Thompson is likely ‘Superstore’, but the creator of the show is not specified...” ▷ <i>Infers the intermediate entity but fails to turn it into an effective retrieval target</i>	Hop 1 Thought: “The documents confirm that Johnny Pemberton appears in the NBC sitcom ‘Superstore’ as the character Bo Thompson.” Prospective Query: “Who created the NBC sitcom ‘Superstore’?”
Subsequent Retrievals: [... , Johnny Pemberton, Johnny Thompson, Bobb'e J. Thompson, Kenan Thompson, Thompson (TV series), Fred Thompson, ...] ▷ <i>Retrieval remains around surface-form Pemberton/Thompson entities and fails to expose the creator evidence</i> Loop remains unresolved across later hops	Hop 2 Retrieval: [... , Johnny Pemberton, Superstore (TV series) , List of Superstore characters, List of Superstore episodes, ...] ▷ <i>Retrieves the answer-bearing page</i> Hop 2 Output: <i>Justin Spitzer</i> Correct answer extracted at Hop 2
Final: ✗ Unresolved (7 retrieval hops)	Final: ✓ Justin Spitzer (2 retrieval hops)

Figure 1: **Retrieval trajectory comparison.** ReSCORE identifies *Superstore* but fails to use it as an effective next-hop retrieval target, causing later retrievals to remain around surface-form Pemberton/Thompson entities. PROBE instead formulates an explicit prospective query for the creator of *Superstore*, retrieves the answer-bearing page, and answers *Justin Spitzer* in two retrieval hops.

tion. To avoid speculative queries contaminating answer generation, PROBE applies *context denoising*, excluding prospective queries from the generation context. To optimize the retriever using self-generated trajectories, we further introduce **Oracle-Based End-Goal Distillation**. Rather than enforcing step-wise supervision, this method distills cross-encoder ranking signals only from the final hop of resolved trajectories, where the query already encodes the accumulated reasoning context and search intent. This provides trajectory-level supervision while preserving early exploration and enables reranker-free inference.

We evaluate PROBE on three multi-hop question answering benchmarks: MuSiQue, 2WikiMultiHopQA, and HotpotQA. PROBE consistently improves both retrieval quality and QA performance over strong iterative baselines, showing that modeling forward-looking search intent with trajectory-level distillation helps discover missing bridging evidence and construct more reliable reasoning paths.

Our key contributions are as follows:

- We propose **PROBE**, which decouples the reasoning state into a *factual summary* and a *prospective query* with context denoising to guide forward-looking retrieval.
- We introduce **Oracle-Based End-Goal Distillation**, transferring cross-encoder signals at the final hop to provide trajectory-level supervision while preserving early exploration.

- Trained from **15K** HotpotQA source questions, PROBE generalizes to unseen benchmarks and achieves strong performance without dataset-specific retriever training.

2 Related Work

Iterative Retrieval and Reasoning Chains. Multi-hop Question Answering (MHQA) necessitates retrieving evidence chains that are not immediately accessible via simple similarity matching (Xiong et al., 2021b; Yang et al., 2018). To address this, the dominant paradigm has shifted towards iterative Retrieval-Augmented Generation (RAG), where an LLM actively interacts with a retriever. Seminal works like IRCot (Trivedi et al., 2023) and Iter-Retgen (Shao et al., 2023) interleave reasoning steps with retrieval, using the LLM to formulate queries based on partial context. FLARE (Jiang et al., 2023) further introduces forward-looking retrieval by predicting upcoming content and retrieving evidence for low-confidence spans. Building on this, agentic frameworks such as Self-RAG (Asai et al., 2024), Adaptive-RAG (Jeong et al., 2024), and Adaptive-Note (Wang et al., 2024) introduce inference-time mechanisms—including dynamic routing, self-reflection, and note-taking—to filter noise and manage retrieval termination. Recent work has further strengthened multi-step retrieval control through explicit planning and evidence management. STRIDE (Chen et al., 2026) co-

ordinates dependent retrieval and reasoning steps through explicit planning and supervision, while PAR²-RAG (Li et al., 2026) combines broad evidence exploration with subsequent refinement under evidence-sufficiency control. A parallel line of work trains the LLM itself via reinforcement learning to adaptively decide when and what to retrieve, including Search-R1 (Jin et al., 2025), ReSearch (Chen et al., 2025) and LeTS (Zhang et al., 2025). Across these approaches, improvements largely come from changing how retrieval is formulated or controlled while leaving the retriever itself fixed. In contrast, our work directly optimizes the retriever for multi-hop reasoning, asking whether stronger retriever training can reduce the reliance on inference-time heuristics.

Dense Retriever Training and Alignment.

Prior work improves dense retrieval through supervised training with labeled query-document pairs (Karpukhin et al., 2020; Xiong et al., 2021a) or unsupervised pretraining (Izacard et al., 2022). More recent methods derive supervision from language models. REPLUG (Shi et al., 2024), ATLAS (Izacard et al., 2023), and ExpandR (Yao et al., 2025) couple retriever learning with LLM-derived objectives. Other work uses LLMs as embedding or ranking models (Zhang et al., 2023; Pradeep et al., 2023); Intermediate Distillation (Li et al., 2024) transfers ranking signals from black-box LLMs, while Syntriever (Kim and Baek, 2025) uses LLM-synthesized data and ranking preferences for retriever training. However, these methods are not designed for iterative multi-hop retrieval, where supervision targets evolve with the reasoning trajectory and are therefore path-dependent. In MHQA, ReSCORE (Lee et al., 2025) derives step-wise retriever supervision from LLM generative likelihoods, scoring documents by their ability to generate the original question and final answer. Such supervision can favor answer-bearing documents while undervaluing bridging evidence that supports later reasoning. PROBE instead applies cross-encoder supervision only at the final hop, where the information need is more fully specified, providing trajectory-level alignment without constraining intermediate exploration.

3 Methodology

3.1 Problem Formulation and Overview

Given a multi-hop question Q and a document corpus $\mathcal{D} = \{d_1, \dots, d_M\}$, the goal is to produce an

exact answer A by reasoning over a chain of evidence. We adopt the *iterative RAG* setting (Trivedi et al., 2023; Lee et al., 2025), where at each hop i , the retriever selects candidate documents from $\mathcal{D}$ conditioned on the current reasoning state, which is updated by integrating the retrieved evidence. This process repeats during iterative reasoning until the generator produces a final answer or a maximum of N thought-generation rounds is reached.

Within this setting, a central challenge is how to train a dense retriever that remains highly effective in complex multi-hop scenarios. PROBE addresses this objective with a two-stage pipeline, illustrated in Figure 2. In the first stage (Section 3.2), we construct training trajectories by guiding the reasoning process to both consolidate verified evidence and anticipate the next missing piece of information. In the second stage (Section 3.3), we train a lightweight Bi-Encoder on these trajectories via **Oracle-Based End-Goal Distillation**, transferring cross-encoder signals exclusively at the final hop to preserve early exploratory freedom. At inference time, PROBE operates efficiently using only the tuned Bi-Encoder and reasoning module, requiring no expensive reranking.

3.2 Prospective Query-Guided Reasoning with Context Denoising

Following ReSCORE (Lee et al., 2025), we construct training data by generating reasoning trajectories over the training set without manual annotation, where the model iteratively extracts verified evidence at each hop and appends it to the next query. However, a key limitation of this design is that the reasoning state only summarizes discovered evidence without anticipating what information is still missing. Consequently, queries remain anchored to accumulated context rather than exploring bridging entities. PROBE addresses this issue by separating the reasoning state into two distinct components. A *factual summary* s_i consolidates verified evidence for answer generation, while a *prospective query* q_i^{next} anticipates the next missing information to guide retrieval.

Thought State Formulation. At hop i , the system maintains a thought state $T_{i-1} = \{t_1, \dots, t_{i-1}\}$ ($T_0 = \emptyset$), where each thought t_j is a structured tuple:

$$t_j = \langle s_j, q_j^{\text{next}} \rangle, \quad j = 1, \dots, i-1 \quad (1)$$

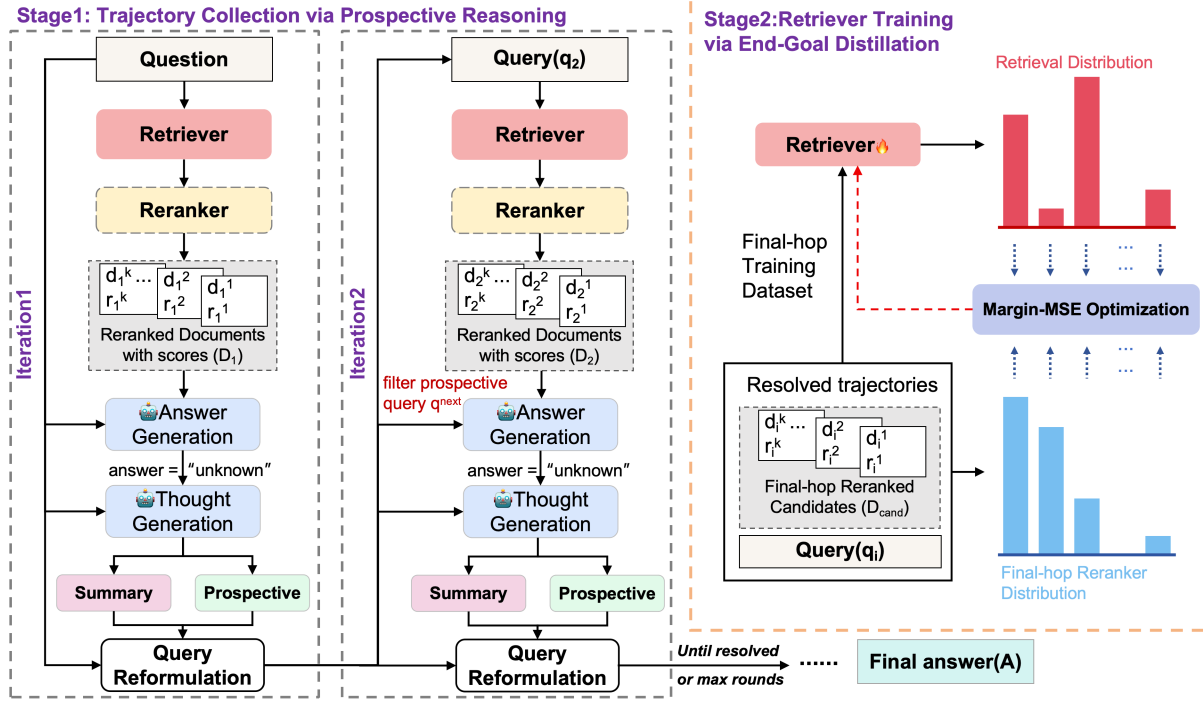


Figure 2: **Overview of the PROBE framework.** **Stage 1 (Trajectory Collection):** During iterative reasoning, the LLM decouples its thoughts into a factual *Summary* and an exploratory *Prospective* query. To prevent speculative search intent from contaminating answer generation, prospective queries are excluded from the reader context. **Stage 2 (End-Goal Distillation):** Using resolved trajectories, the dense retriever is optimized via Margin-MSE to distill Cross-Encoder ranking signals *exclusively at the final hop*. At inference time, the Reranker is completely discarded, yielding an efficient, rerank-free pipeline.

Here $s_j \in \mathcal{S}$ is a summary of facts verified from the documents retrieved at hop j , and $q_j^{\text{next}} \in \mathcal{Q}$ is a self-contained search directive that anticipates the next missing piece of bridging information. Critically, these two components are generated jointly in a single LLM pass but consumed by different downstream modules, as described next.

Step 1: Retrieval. At each hop, we retrieve documents using a search query. At the first hop ($i = 1$), the query is simply the original question Q . For later hops ($i > 1$), the query concatenates Q with all previously generated components, yielding

$$q_i^{\text{search}} = [Q \parallel s_1 \parallel q_1^{\text{next}} \parallel \dots \parallel s_{i-1} \parallel q_{i-1}^{\text{next}}] \quad (2)$$

where $\parallel$ denotes string concatenation. This design ensures that historical summaries anchor the search in verified facts, while the latest prospective query q_{i-1}^{next} explicitly targets missing evidence. The Bi-Encoder first retrieves a candidate pool $\mathcal{C}_i \subset \mathcal{D}$, which is reranked by a Cross-Encoder. We retain a reranked candidate set $\tilde{\mathcal{C}}_i \subseteq \mathcal{C}_i$ together with its relevance scores for subsequent distillation, while a smaller top-ranked subset $D_i \subseteq \tilde{\mathcal{C}}_i$ is provided to the LLM for answer and thought generation.

Step 2: Context Denoising for Answer Generation. Given D_i , the LLM attempts to answer Q . However, directly using the full thought state as context risks exposing speculative hypotheses to the generator. We therefore filter out all prospective queries, retaining only fact-grounded summaries:

$$\hat{A}_i = \text{LLM}(Q, \{s_1, \dots, s_{i-1}\}, D_i) \quad (3)$$

where $\hat{A}_i$ is either a predicted answer string or *Unknown*. If the reader produces a non-*Unknown* answer, the trajectory terminates; otherwise, the system proceeds to thought generation and the next retrieval round.

Step 3: Thought Generation and Query Reformulation. When *Unknown* is returned, the LLM generates the next structured thought:

$$\langle s_i, q_i^{\text{next}} \rangle = \text{LLM}(Q, T_{i-1}, D_i) \quad (4)$$

The thought state is updated as $T_i = T_{i-1} \cup \{t_i\}$, and the process loops back to Step 1 with q_i^{next} driving the next retrieval hop, continuing until the LLM produces a final answer or the maximum of N thought-generation rounds is reached.

3.3 Oracle-Based End-Goal Distillation

Given the collected trajectories, the second stage trains a lightweight Bi-Encoder retriever to enable efficient inference. In multi-hop retrieval, intermediate supervision can be unreliable because early queries often capture only partial information needs. Furthermore, relevance scores derived from rerankers or LLMs may struggle to consistently determine the utility of a document at these early stages, especially when intermediate targets are ambiguous and path-dependent. By contrast, the final hop I incorporates the accumulated reasoning context and search intent along the trajectory, yielding a more fully specified information need. We therefore apply Cross-Encoder supervision exclusively at this final stage. By utilizing q_I^{search} as the trajectory-aware query, this approach shifts the optimization paradigm from noisy step-wise imitation to trajectory-level alignment.

Teacher Signal Construction. For each resolved trajectory, we extract teacher signals exclusively from its final hop I . We define the distillation candidate set as $D_{\text{cand}} = \tilde{C}_I$, i.e., the reranked candidate pool retained during trajectory collection. The corresponding precomputed Cross-Encoder scores are directly used as teacher relevance scores:

$$r_t(q_I^{\text{search}}, d) = \text{CrossEncoder}(q_I^{\text{search}} \oplus d) \quad (5)$$

where $\oplus$ denotes sequence concatenation, t denotes the teacher model, and d denotes a document. To filter ambiguous signals, we retain only instances where the score margin between the top-1 (d^+) and top-2 (d^-) documents within D_{cand} exceeds a threshold ϵ , requiring

$$r_t(q_I^{\text{search}}, d^+) - r_t(q_I^{\text{search}}, d^-) \geq \epsilon \quad (6)$$

Margin-MSE Distillation. The student Bi-Encoder defines relevance as

$$r_s(q_I^{\text{search}}, d) = E_Q(q_I^{\text{search}})^\top E_D(d) \quad (7)$$

where $r_s(\cdot)$ denotes the student relevance computed via inner product, where s denotes the student model, and E_Q, E_D are the query and document encoders. To transfer continuous ranking preferences, we define the score margin for any document pair (d_u, d_v) under model $m \in \{s, t\}$ as

$$\delta_m(d_u, d_v) = r_m(q_I^{\text{search}}, d_u) - r_m(q_I^{\text{search}}, d_v) \quad (8)$$

over the set of document pairs $\mathcal{P}$ formed from D_{cand} , yielding

$$\mathcal{L} = \frac{1}{|\mathcal{P}|} \sum_{(d_u, d_v) \in \mathcal{P}} (\delta_s(d_u, d_v) - \delta_t(d_u, d_v))^2 \quad (9)$$

At inference time, the trained Bi-Encoder is applied at every reasoning hop to iteratively retrieve candidate documents. Despite being trained only with final-hop supervision, the retriever generalizes well to intermediate hops, as final-hop queries capture a more fully specified representation of the information need. Consequently, the learned relevance function can transfer to earlier, partially specified hops while avoiding direct optimization on noisy and ambiguous intermediate supervision. We validate this empirically in Section 4.3.

4 Experiments

4.1 Settings

Datasets. We conduct experiments on three popular multi-hop question answering (MHQA) benchmarks: **HotpotQA** (Yang et al., 2018), **MuSiQue** (Trivedi et al., 2022), and **2WikiMulti-HopQA** (Ho et al., 2020). Each dataset contains questions that require composing evidence across multiple documents, making them suitable for evaluating iterative retrieval and multi-hop reasoning. Following prior iterative-RAG setups (Trivedi et al., 2023; Jeong et al., 2024; Lee et al., 2025), we run evaluation on subsampled versions of the validation sets and their corresponding retrieval databases. Importantly, while these datasets provide ground-truth (GT) supporting document labels, we do *not* use GT labels for training our retriever.

Baselines. We compare PROBE against baselines from two categories. **Methods without retriever training** rely on inference-time strategies to improve iterative retrieval and reasoning, including ReAct (Yao et al., 2023), FLARE (Jiang et al., 2023), Self-RAG (Asai et al., 2024), Adaptive-Note (Wang et al., 2024), IRCOT (Trivedi et al., 2023), Adaptive-RAG (Jeong et al., 2024), and STRIDE (Chen et al., 2026), which explicitly coordinates dependent retrieval and reasoning steps. Search-R1 (Jin et al., 2025) further trains the LLM via reinforcement learning to adaptively issue search queries. **Methods with retriever training** are represented by ReSCORE (Lee et al., 2025), which provides step-wise supervision using LLM likelihoods to score intermediate documents by

Method	MuSiQue		HotpotQA		2WikiMHQA		AVG	
	EM	F1	EM	F1	EM	F1	EM	F1
<i>Methods without Retriever Training</i>								
ReAct (GPT-3.5+BM25) [†]	10.2	19.7	36.0	46.9	28.0	37.3	24.7	34.6
FLARE (GPT-3.5+BM25) [†]	11.2	18.7	36.4	47.8	31.8	42.8	26.5	36.4
Self-RAG (GPT-3.5+BM25) [†]	10.6	19.2	33.8	44.4	24.4	30.8	22.9	31.5
Adaptive-Note (GPT-3.5+BM25) [†]	13.2	24.2	45.6	58.4	43.2	54.2	34.0	45.6
IRCoT (Flan-T5-XL+BM25) [†]	22.0	31.8	44.4	56.2	49.7	54.9	38.7	47.6
Adaptive-RAG (Flan-T5-XL+BM25) [†]	23.6	31.8	42.0	53.8	40.6	49.8	35.4	45.1
Search-R1 (Qwen2.5-7B+Contriever) [‡]	24.4	<u>34.0</u>	<u>52.4</u>	<u>64.9</u>	54.4	63.0	<u>43.7</u>	<u>54.0</u>
STRIDE (Qwen3-8B+Contriever)*	<u>24.6</u>	33.6	49.4	59.4	<u>57.0</u>	<u>65.6</u>	<u>43.7</u>	52.9
<i>Methods with Retriever Training</i>								
ReSCORE (Llama-3.1-8B+Contriever, 278K [§]) [†]	23.4	32.7	47.2	59.3	50.0	59.7	40.2	50.6
<i>Our Framework</i>								
Untrained Baseline (Llama-3.1-8B+BM25)	21.2	31.6	45.8	58.7	53.2	60.7	40.1	50.3
Untrained Baseline (Llama-3.1-8B+Contriever)	20.4	30.9	48.0	60.8	52.2	61.7	40.2	51.1
PROBE (trained Contriever, 15K)	25.0	34.5	53.4	66.7	59.2	67.6	45.9	56.3

Table 1: **Comparison with iterative RAG systems on three MHQA benchmarks.** [†] Results reported by ReSCORE (Lee et al., 2025); [‡] reproduced by us using FlashRAG²; * reproduced by us using the official implementation of STRIDE. [§] ReSCORE trains dataset-specific retrievers on the full training sets, whereas PROBE trains a single retriever from 15K HotpotQA source questions. Bold and underline denote the best and second-best results.

their ability to generate both the original question and the final answer.

Metrics. We report Exact Match (EM) and F1 using official answer-level evaluation scripts provided in each dataset. To evaluate retrieval quality within an iterative setting, we additionally report *multi-hop recall* at k ($\text{MHR}_i@k$), defined as the cumulative recall of supporting documents across the first i iterations:

$$\text{MHR}_i@k = \frac{|\mathcal{D}^* \cap \bigcup_{j=1}^i \mathcal{D}_{j,k}|}{|\mathcal{D}^*|}, \quad (10)$$

where $\mathcal{D}^*$ is the set of ground-truth supporting documents and $\mathcal{D}_{j,k}$ denotes the top- k retrieved documents at retrieval hop j .

Implementation Details. We use Contriever (Izacard et al., 2022) pre-trained on MS-MARCO (Nguyen et al., 2016) as the base retriever, and train the question encoder using trajectories collected from a 15K subset of HotpotQA. Relevance supervision is provided by BAAI/bge-reranker-v2-m3 (Chen et al., 2024), which is used only for offline trajectory synthesis and retriever distillation. We use Llama-3.1-8B-Instruct (Team, 2024) as the reader. During inference, PROBE starts with one initial retrieval and allows up to $N = 6$

thought-generation rounds when the reader outputs *Unknown*. Each thought triggers one additional retrieval, resulting in at most $N + 1 = 7$ retrieval calls in total. The Cross-Encoder reranker is entirely removed at inference time. Detailed hyperparameters, computational cost, prompt templates, and trajectory filtering procedures are provided in Appendix A and B.

4.2 Main Results

Comparison with State-of-the-Art Methods. Table 1 compares PROBE with strong iterative retrieval and reasoning baselines. PROBE achieves the best EM and F1 scores across all three MHQA benchmarks, as well as the best average performance. Among **methods without retriever training**, PROBE outperforms both inference-time retrieval strategies such as ReAct, FLARE, IRCoT, and Adaptive-RAG, as well as stronger multi-step reasoning methods such as Search-R1 and STRIDE. These consistent gains suggest that directly adapting the dense retriever to multi-hop information needs provides benefits beyond improving retrieval control with a fixed retriever. Among **methods with retriever training**, PROBE also outperforms ReSCORE, the most directly comparable baseline using automatically derived supervision. This supports our motivation for avoiding step-wise gen-

erative supervision and instead transferring Cross-Encoder ranking signals at the final hop, where the information need is more fully specified. The untrained variants with BM25 and off-the-shelf Contriever show that the prospective reasoning pipeline is already competitive without retriever tuning. Replacing the off-the-shelf Contriever with the distilled retriever further improves performance, yielding the strongest overall results. Since Table 1 follows the original or reproduced configurations of different systems, we provide an effectiveness–efficiency comparison under a controlled evaluation setup, including per-query latency, in Appendix C.6.

4.3 Ablation Study

To validate the effectiveness of our design choices, we conduct ablations focusing on (i) the role-decoupled reasoning components (Factual Summary and Prospective Query), (ii) the impact of end-goal distillation, and (iii) the training signals used for retriever optimization.

Effectiveness of Core Components. Table 2 reports the contribution of each design component in PROBE. Compared with the vanilla iterative RAG baseline, incorporating the **Prospective Query (PQ)** yields consistent improvements across all datasets by restoring the retriever’s ability to explore missing information needs. Further combining PQ with the **Factual Summary (Sum. + PQ)** leads to additional gains on multiple benchmarks, as the summary preserves established context while PQ explicitly models future information needs. On MuSiQue, however, PQ alone slightly outperforms the full Sum. + PQ pipeline. We attribute this to its compositional design, where exact entity mentions can be beneficial for subsequent retrieval, and summarization may occasionally smooth such signals. Overall, the full pipeline remains the most robust configuration across benchmarks.

Distillation Strategy. As shown in Table 2, strategies incorporating intermediate-hop supervision (Inter-hop and All-hop) generally achieve the highest retrieval recall (MHR), yet this does not translate into better QA performance. On HotpotQA, where two-hop chains are relatively short and intermediate targets closely resemble the final retrieval objective, Inter-hop distillation performs comparably to Final-hop. However, on MuSiQue

Components			QA		MHR _i @8	
Sum.	PQ	Dist.	EM	F1	<i>i</i> = 2	<i>i</i> = final
MuSiQue						
✗	✗	✗	11.2	18.0	48.0	48.7
✓	✗	✗	19.0	28.5	58.5	59.3
✗	✓	✗	21.2	31.1	57.6	59.1
✓	✓	✗	20.4	30.9	60.5	61.5
✓	✓	Inter-hop	23.2	32.8	69.2	69.8
✓	✓	Final-hop	25.0	34.5	67.5	68.3
✓	✓	All-hop	24.6	34.3	69.7	70.0
HotpotQA						
✗	✗	✗	36.0	49.0	64.2	64.9
✓	✗	✗	44.0	58.1	76.8	77.4
✗	✓	✗	45.2	57.9	71.3	71.5
✓	✓	✗	48.0	60.8	78.3	79.0
✓	✓	Inter-hop	53.8	66.1	86.8	87.3
✓	✓	Final-hop	53.4	66.7	85.8	85.9
✓	✓	All-hop	51.8	64.7	86.8	86.9
2WikiMultiHopQA						
✗	✗	✗	35.4	42.0	60.7	62.3
✓	✗	✗	44.0	52.5	75.9	76.3
✗	✓	✗	45.0	53.9	75.8	77.1
✓	✓	✗	52.2	61.7	79.5	80.4
✓	✓	Inter-hop	54.4	62.9	92.2	92.2
✓	✓	Final-hop	59.2	67.6	90.6	90.9
✓	✓	All-hop	56.4	64.5	93.0	93.0

Table 2: **Ablation study on PROBE’s core components.** **Sum.** and **PQ** denote the Factual Summary and Prospective Query modules respectively. **Dist.** indicates the retriever distillation strategy: **Inter-hop** supervises only intermediate hops, **Final-hop** supervises only the final hop, and **All-hop** combines distillation signals from both intermediate and final hops.

and 2WikiMHQA, where reasoning chains are longer and admit multiple valid intermediate paths, Inter-hop distillation becomes suboptimal, degrading both EM and F1 relative to Final-hop distillation. We attribute this to the ambiguous and path-dependent nature of intermediate supervision. Complex questions often admit multiple valid reasoning trajectories, making intermediate retrieval targets non-unique. Supervising the retriever on specific intermediate documents therefore encourages it to overfit to a single reasoning path, introducing noisy signals that are weakly aligned with the underlying information need. In contrast, final-hop distillation avoids directly supervising these ambiguous intermediate targets. At the final hop, the accumulated reasoning context and search intent more fully specify the information need, enabling the Cross-Encoder to provide a more stable ranking signal. This yields trajectory-level supervision without committing the retriever to a particular intermediate retrieval target. All-hop distillation partially mitigates this by combining both signals, yet residual interference from intermediate

²<https://github.com/RUC-NLPIR/FlashRAG>

Label	QA		MHR _i @8		
	EM	F1	$i = 1$	$i = 2$	$i = \text{final}$
MuSiQue					
None	20.4	30.9	48.0	60.5	61.5
GT	23.0	32.7	46.0	57.8	58.7
$P_{\text{LM}}(a q, d)$	22.0	31.8	47.4	61.4	62.4
PROBE	25.0	34.5	51.6	67.5	68.3
HotpotQA					
None	48.0	60.8	64.2	78.3	79.0
GT	49.2	62.2	63.5	78.9	79.8
$P_{\text{LM}}(a q, d)$	49.8	62.6	60.5	77.1	77.1
PROBE	53.4	66.7	68.4	85.8	85.9
2WikiMHQA					
None	52.2	61.7	60.7	79.5	80.4
GT	56.2	64.6	63.9	85.1	85.6
$P_{\text{LM}}(a q, d)$	52.2	60.0	61.1	82.4	82.7
PROBE	59.2	67.6	66.3	90.6	90.9

Table 3: **Comparisons of different supervision signals for fine-tuning retrievers on three MHQA benchmarks.** **None** denotes the base retriever without additional fine-tuning. **GT** uses binary relevance labels. $P_{\text{LM}}(a | q, d)$ uses soft labels derived from the LLM’s conditional answer likelihood given a question–document pair. **PROBE** uses soft relevance scores produced by a Cross-Encoder reranker.

hops still degrades QA accuracy on most benchmarks. These results suggest that high intermediate retrieval recall alone does not guarantee better QA performance; what matters more is retrieving the right evidence when the information need becomes sufficiently complete.

Impact of Supervision Signals. Table 3 validates the choice of supervision signal for retriever training. PROBE uses Cross-Encoder (CE) reranking scores over the trajectory-aware query q_I^{search} , providing graded relevance signals for fine-grained document ranking. In contrast, GT supervision treats all annotated supporting documents as positives for the original question, ignoring their hop-specific roles in the reasoning chain. This introduces supervision ambiguity in MHQA, where different evidence documents may correspond to different intermediate information needs. Generation-based soft labels from LLM likelihoods are also noisy, as they can conflate document relevance with the generator’s calibration and parametric knowledge. By distilling trajectory-aware CE scores, PROBE provides a cleaner retrieval-oriented signal and achieves stronger performance than these alternatives. Notably, despite using supervision only from the final hop, PROBE also improves MHR at

earlier retrieval hops over the untrained retriever, suggesting that the learned relevance preferences remain useful for partially specified intermediate queries.

4.4 Generality of the Distilled Retriever

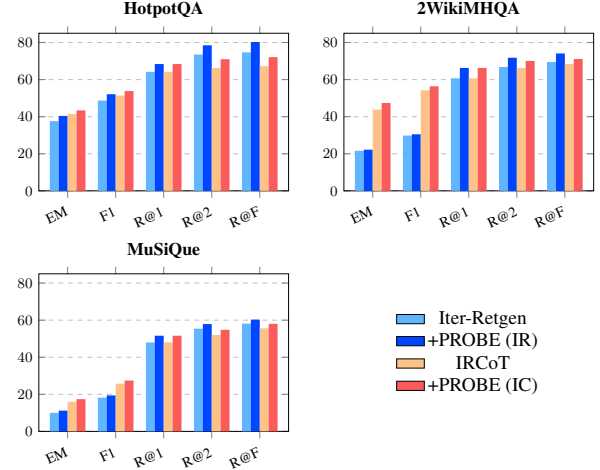


Figure 3: **Generality analysis.** EM and F1 denote QA performance, while $R@i$ denotes multi-hop retrieval recall (MHR_i@8) at hop i . $R@F$ denotes MHR@8 at the final retrieval step.

To evaluate whether the distilled retriever generalizes across different iterative RAG pipelines, we integrate it into several strong baselines, including **Iter-Retgen** and **IRCoT**. All methods are re-implemented using Llama 3.1 and Contriever within the FlashRAG framework, with corresponding hyperparameters detailed in section A. To isolate the retrieval impact, we instantiate these baselines with an untrained Contriever and compare them against our distilled model, leaving the original control logic unchanged. As shown in Figure 3, substituting the standard retriever with our model consistently improves both retrieval recall and downstream QA performance across all baselines. These results suggest that the relevance signals learned through our training objective transfer robustly across different reasoning frameworks within the same corpus, indicating strong within-domain generalization rather than reliance on a specific reasoning pipeline.

We provide additional evidence in Appendix C and Appendix D to further assess the robustness, efficiency, and limitations of PROBE. We evaluate single-hop QA generalization and study query formulation strategies to isolate the effect of prospective queries. We further examine recovery from imperfect early retrieval states, as well as query

truncation and sensitivity to the distillation margin threshold. A controlled effectiveness–efficiency comparison covers closed-book, single-pass RAG, and iterative retrieval baselines, together with an analysis of rerank-free inference. We additionally evaluate generalization across different LLM backbones. Finally, we provide case-level failure and trajectory analyses to characterize remaining errors and discuss future directions.

5 Conclusion

In this paper, we propose PROBE, a retriever-training framework that uses prospective retrieval to adapt dense retrievers to multi-hop question answering. By decoupling the reasoning state into a factual summary and a prospective query, PROBE separates exploratory search intent from grounded answer generation. To train the retriever without suppressing early exploration, we introduce Oracle-Based End-Goal Distillation, which transfers cross-encoder ranking signals exclusively at the final hop to provide trajectory-level supervision. Extensive experiments on three MHQA benchmarks demonstrate that PROBE outperforms prior methods by consistent margins. We hope our work inspires future research on retriever alignment for iterative reasoning systems.

Limitations

While PROBE demonstrates consistent improvements across three standard MHQA benchmarks, its evaluation remains limited by the current benchmark setting. Most existing MHQA datasets are built from curated Wikipedia-style corpora with well-formed questions, which may not fully capture real-world multi-hop information-seeking scenarios involving noisy heterogeneous sources, underspecified queries, evolving documents, and domain-specific terminology. Thus, although PROBE is effective under widely used MHQA protocols, evaluating iterative retrieval methods on more realistic multi-hop benchmarks remains an important direction for future work. PROBE also introduces additional inference complexity compared with single-pass retrieval systems, since each reasoning hop requires an independent retrieval call. Moreover, its offline trajectory synthesis stage relies on a powerful LLM and Cross-Encoder, resulting in additional pre-deployment cost. Future work may explore more efficient synthesis strategies, trajectory correction mechanisms, and benchmarks that better

reflect practical multi-hop retrieval scenarios.

Ethics Statement

This study uses publicly available QA benchmarks, including MuSiQue, 2WikiMultiHopQA, HotpotQA, Natural Questions, TriviaQA, and SQuAD. We follow the dataset-specific licenses and terms of use for all datasets and pretrained model artifacts used in this work. All resources are used only for research-oriented training and evaluation of retrieval-augmented question answering. We do not collect new human-subject data or perform deanonymization. For personally identifying or offensive content, we rely on the public curation and documentation of these established QA benchmarks, while acknowledging that web- or Wikipedia-derived resources may still contain biases, outdated information, or other problematic content. We ensure consistency in training and evaluation conditions to support fair comparisons. Nevertheless, the underlying language models and retrievers may still propagate biases or hallucinate incorrect answers when exposed to incomplete or unreliable evidence. We therefore view PROBE as a research system rather than as a deployable system for high-stakes applications.

Acknowledgments

This work was supported by the Noncommunicable Chronic Diseases-National Science and Technology Major Project (Grant No. 2026ZD0558202). Haobo Wang is also supported by the Ningbo Yongjiang Talent Introduction Programme (No. 2023A-399-G). We also acknowledge the Zhejiang University College of Computer Science and Technology–Zhejiang Geely Holding Group AI Joint Research Laboratory for facilitating this collaborative research.

References

- Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *ICLR*.
- Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. *CoRR*, abs/2402.03216.
- Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen

- Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. 2025. ReSearch: Learning to Reason with Search for LLMs via Reinforcement Learning. *CoRR*, abs/2503.19470.
- Wei Chen, Lili Zhao, Zhi Zheng, Huijun Hou, and Tong Xu. 2026. STRIDE: Strategic Iterative Decision-Making for Retrieval-Augmented Multi-Hop Question Answering. In *SIGIR*, pages 202–212.
- Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise Zero-Shot Dense Retrieval without Relevance Labels. In *ACL*, pages 1762–1777.
- 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 *COLING*, pages 6609–6625.
- Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *Trans. Mach. Learn. Res.*
- Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot Learning with Retrieval Augmented Language Models. *J. Mach. Learn. Res.*, pages 251:1–251:43.
- Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024. Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. In *NAACL*, pages 7036–7050.
- Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active Retrieval Augmented Generation. In *EMNLP*, pages 7969–7992.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning. *CoRR*, abs/2503.09516.
- Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *ACL*, pages 1601–1611.
- Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP*, pages 6769–6781.
- Minsang Kim and Seung Jun Baek. 2025. Syntriever: How to Train Your Retriever with Synthetic Data from LLMs. In *NAACL Findings*, pages 2523–2539.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: a Benchmark for Question Answering Research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.
- Dosung Lee, Wonjun Oh, Boyoung Kim, Minyoung Kim, Joonsuk Park, and Paul Hongsuck Seo. 2025. ReSCORE: Label-free Iterative Retriever Training for Multi-hop Question Answering with Relevance-Consistency Supervision. In *ACL*, pages 341–359.
- Xingyu Li, Rongguang Wang, Yuying Wang, Mengqing Guo, Chenyang Li, Tao Sheng, Sujith Ravi, and Dan Roth. 2026. PAR²-RAG: Planned Active Retrieval and Reasoning for Multi-Hop Question Answering. *CoRR*, abs/2603.29085.
- Zizhong Li, Haopeng Zhang, and Jiawei Zhang. 2024. Intermediate Distillation: Data-Efficient Distillation from Black-Box LLMs for Information Retrieval. *CoRR*, abs/2406.12169.
- Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MACHINE Reading Comprehension Dataset. In *NeurIPS*, volume 1773.
- Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. RankVicuna: Zero-Shot Listwise Document Reranking with Open-Source Large Language Models. *CoRR*, abs/2309.15088.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *EMNLP*, pages 2383–2392.
- Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*, pages 109–126.
- Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy. In *EMNLP Findings*, pages 9248–9274.
- Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-Augmented Black-Box Language Models. In *NAACL*, pages 8371–8384.
- Llama Team. 2024. The Llama 3 Herd of Models. *CoRR*, abs/2407.21783.

- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop Questions via Single-hop Question Composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. In *ACL*, pages 10014–10037.
- Ruobing Wang, Daren Zha, Shi Yu, Qingfei Zhao, Yuxuan Chen, Yixuan Wang, Shuo Wang, Yukun Yan, Zhenghao Liu, Xu Han, Zhiyuan Liu, and Maosong Sun. 2024. Retriever-and-Memory: Towards Adaptive Note-Enhanced Retrieval-Augmented Generation. *CoRR*, abs/2410.08821.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*.
- Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021a. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *ICLR*.
- Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2021b. Answering Complex Open-Domain Questions with Multi-Hop Dense Retrieval. In *ICLR*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *EMNLP*, 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 *ICLR*.
- Sijia Yao, Pengcheng Huang, Zhenghao Liu, Yu Gu, Yukun Yan, Shi Yu, and Ge Yu. 2025. ExpandR: Teaching Dense Retrievers Beyond Queries with LLM Guidance. In *EMNLP*, pages 19036–19054.
- Qi Zhang, Shouqing Yang, Lirong Gao, Hao Chen, Xiaomeng Hu, Jinglei Chen, Jiexiang Wang, Sheng Guo, Bo Zheng, Haobo Wang, and Junbo Zhao. 2025. LeTS: Learning to Think-and-Search via Process-and-Outcome Reward Hybridization. In *EMNLP*, pages 5109–5122.
- Xin Zhang, Zehan Li, Yanzhao Zhang, Dingkun Long, Pengjun Xie, Meishan Zhang, and Min Zhang. 2023. Language Models are Universal Embedders. *CoRR*, abs/2310.08232.

A Experimental Details and Hyperparameters

Hyperparameters for the baseline methods follow their original papers or accompanying codebases whenever possible, with minor adjustments for stable reproduction. Table 4 reports the final configurations. For PROBE, we keep the iterative settings consistent with the compared methods unless otherwise specified.

In Table 4, **Max Tokens** denotes the maximum generation length, **Temperature** (T) denotes the decoding temperature, with method-specific values following the corresponding configurations, **Top- k** is the number of retrieved documents provided at each iteration, **Max Step** defines the maximum number of thought-generation rounds after the initial retrieval, and **Max Fail** is the retry limit for invalid model outputs.

A.1 Datasets and Evaluation Metrics

We evaluate on three Wikipedia-based MHQA benchmarks: **HotpotQA** (Yang et al., 2018), **2WikiMultiHopQA** (Ho et al., 2020), and **MuSiQue** (Trivedi et al., 2022). Following prior iterative-RAG evaluations (Trivedi et al., 2023; Jeong et al., 2024; Lee et al., 2025), we use the original Wikipedia corpus for HotpotQA and construct corpora from the provided contexts for 2WikiMultiHopQA and MuSiQue. For each MHQA dataset, we tune hyperparameters on 100 randomly sampled development examples and evaluate on another 500 randomly sampled examples. For supplementary single-hop QA evaluation, we further use **Natural Questions** (Kwiatkowski et al., 2019), **TriviaQA** (Joshi et al., 2017), and **SQuAD** (Rajpurkar et al., 2016). We follow the same evaluation protocol and evaluate each single-hop dataset on 500 randomly sampled examples. We report Exact Match (EM) and F1 for answer quality. For multi-hop retrieval, we additionally report Multi-Hop Recall (MHR), which measures the cumulative coverage of gold supporting documents across the reasoning trajectory.

A.2 Training Hyperparameters

We fine-tune only the question encoder of Contriever and keep the document encoder frozen. To improve the quality of distillation signals, we discard training instances whose top-pair Cross-Encoder score margin is below $\epsilon = 0.0001$. The retriever is optimized with AdamW using a learn-

	Max Tokens	T	Top k	Max Step	Max Fail
FLARE	256	0.	8	-	-
Self-RAG	128	0.	8	6	-
Adaptive-Note	128	0.1	5	3	1
Iter-Retgen	128	0.	8	6	-
IRCoT	128	0.	8	6	-
ReSCORE	64	0.	8	6	-
Search-R1	1024	0.	3	6	-
STRIDE	512	0.8	3	5	2
PROBE	128	0.	8	6	-

Table 4: Hyperparameters used for reproducing each method.

ing rate of 1×10^{-5} , weight decay 0.01, batch size 32, 8 gradient accumulation steps, warmup ratio 0.1, and gradient clipping norm 1.0. We train for 20 epochs.

A.3 Computational Cost

PROBE incurs offline cost for trajectory synthesis and retriever distillation, but these components are not used during deployment. Starting from 15K HotpotQA training questions, trajectory synthesis and filtering take approximately 15 hours using two RTX 3090 GPUs for LLM inference and one additional GPU for retrieval/reranking. Retriever distillation takes 147 minutes on a single RTX 3090 with 20 epochs, batch size 32, and 8 gradient accumulation steps.

Table 5 summarizes the cost breakdown. At inference time, PROBE directly uses the distilled bi-encoder retriever and the reader model, without trajectory synthesis, Cross-Encoder scoring, self-reflection, or branch-level aggregation. Thus, PROBE trades a moderate one-time offline cost for a lightweight online pipeline.

B Details of the PROBE Framework

This section provides implementation details for PROBE, including the inference prompts, trajectory filtering, and retriever distillation setup.

B.1 Inference and Reasoning Prompts

PROBE uses two prompts during iterative reasoning. The **Thought Generation Prompt** is invoked when the current evidence is insufficient to answer the question. It takes the original question, previous thoughts, and newly retrieved documents as input, and outputs a structured thought containing a **Factual Summary** and a **Prospective Query** (q^{next}). The summary records verified evidence, while the prospective query specifies the

Stage	Resource	Time / Scale	Inference
Silver trajectory synthesis	2 RTX 3090 for LLM inference; 1 GPU for retrieval/reranking	15K initial questions; ~ 15 hours	No
Retriever distillation	1 RTX 3090	147 minutes; 20 epochs	No
PROBE inference	2 RTX 3090 for LLM inference; 1 GPU for retrieval	3.27s/query on average	Yes

Table 5: **Computational cost of PROBE.** Offline synthesis and distillation are performed once before deployment.

next missing information to retrieve. The **Answer Generation Prompt** determines whether the retrieved evidence is sufficient for answering. To avoid speculative information affecting the final prediction, PROBE removes prospective queries from the answer-generation context and only keeps retrieved documents and factual summaries. If the reader deems the evidence sufficient, it outputs a final answer; otherwise, it returns *Unknown* and continues to the next retrieval step. Full prompts are shown in Figure 4 and Figure 5.

B.2 Training Trajectory Collection and Filtering

We sample 15K HotpotQA training questions and run the PROBE reasoning pipeline without manual annotation, producing 31,396 raw trajectories. We then keep only trajectories that reach a non-Unknown final answer, yielding 14,770 resolved trajectories. Finally, we apply the Cross-Encoder margin filter with $\epsilon = 0.0001$, retaining 12,738 training instances for retriever distillation.

B.3 Oracle-Based End-Goal Distillation Setup

Unlike prior step-wise frameworks that rely on noisy LLM generation probabilities for intermediate retrieval supervision, PROBE utilizes an offline Cross-Encoder to provide highly accurate, margin-based relevance signals. For the offline distillation phase, we use a powerful Cross-Encoder (BGE-Reranker-v2-M3) as the reranking teacher.

The distillation setup involves two key components.

- *Cross-Encoder Scoring.* At each retrieval hop, Contriever/FAISS first retrieves the top-128 candidates. BGE-Reranker-v2-M3 reranks this pool and retains the top 32 documents together with their scores. The top 8 reranked documents are provided to the LLM for answer and thought generation, while all 32 documents are stored for subsequent retriever distillation. At the final resolved hop, these 32 documents constitute D_{cand} .

- *Margin-MSE Optimization.* For each final-hop query, we compute pairwise score differences over the retained $M = 32$ candidate documents using the teacher model. The student Bi-Encoder is then optimized via Mean Squared Error to match these pairwise teacher margins.

By applying this distillation **exclusively at the final hop**, PROBE successfully transfers trajectory-level ranking capabilities to the dense retriever without penalizing the exploratory nature of early-hop prospective queries.

C Additional Experimental Results

This section reports supplementary evaluations of PROBE’s design choices, robustness, efficiency, and generalization.

C.1 Generalization Beyond Multi-Hop QA

The main experiments focus on iterative MHQA, but we also evaluate whether PROBE’s retriever training harms single-hop QA performance. We test on Natural Questions (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and SQuAD (Rajpurkar et al., 2016), using the same open-domain evaluation setting for all compared methods.

For this evaluation, we use the corresponding implementations integrated in FlashRAG³. LLM inference is served with vLLM⁴, processing one query at a time to avoid request-level batching effects. Unless otherwise specified, we use two RTX 3090 GPUs for LLM inference and one additional RTX 3090 GPU for retrieval or reranking. For SELF-RAG and Search-R1, we use the trained checkpoints released or specified by their original papers and run them through the corresponding implementations integrated in FlashRAG. IRCot also uses the implementation integrated in FlashRAG. PROBE is rerun under the same setup.

³<https://github.com/RUC-NLPIR/FlashRAG>

⁴<https://github.com/vllm-project/vllm>

Table 6 distinguishes two PROBE variants. **PROBE trained** uses the distilled Contriever trained on 12,738 retained final-hop instances generated from 15K HotpotQA source questions, while **PROBE untrained** keeps the same reasoning and query formulation pipeline but replaces the retriever with off-the-shelf Contriever.

As shown in Table 6, PROBE trained is consistently comparable to or better than PROBE untrained. The gains are small on NQ and TriviaQA, where one retrieval step is often sufficient, but clearer on SQuAD. These results suggest that final-hop distillation does not sacrifice single-hop QA ability, although PROBE’s primary advantage remains in iterative multi-hop retrieval.

C.2 Effect of Query Formulation Strategies

Table 7 compares different query reformulation strategies. Naive query rewriting (**LLM-rewrite**) is prone to semantic drift because fluent rewritten queries may over-commit to unverified hypotheses from partial evidence. **Thought-concat** reduces this risk by retaining the original question in later queries, but long concatenated contexts can dilute the search intent. PROBE performs best by maintaining a fact-grounded summary while generating an explicit next-hop query for the remaining information need. Context denoising further prevents speculative query content from entering answer generation.

C.3 Recovery from Imperfect Retrieval States

Final-hop distillation is learned from resolved trajectories, whereas inference may encounter incomplete or misleading evidence states at earlier hops. This creates a potential train–inference mismatch: although supervision is obtained only from final-hop queries, the trained retriever must still operate on partially specified intermediate queries. We therefore examine whether PROBE can recover after an imperfect first-hop retrieval rather than requiring the reasoning trajectory to remain on an ideal path.

For each question, we compute cumulative MHR@8 as the fraction of gold supporting documents contained in the union of top-8 results retrieved up to hop i . We focus on trajectories whose first hop does not recover the complete supporting set. We define **retrieval recovery** as any increase in cumulative MHR@8 after the incomplete first hop. **Full-chain recovery** requires the final cumulative MHR@8 to reach 1, while **semantic answer**

recovery additionally requires the recovered trajectory to produce a semantically correct final answer. Together, these criteria distinguish partial evidence recovery, complete supporting-chain reconstruction, and recovery that ultimately leads to a correct answer.

As shown in Table 8, PROBE recovers additional supporting evidence in 62.7% of initially incomplete trajectories and reconstructs the complete supporting chain in 50.9% of them. Its semantic answer recovery rate reaches 37.1%, indicating that the recovered evidence also contributes to successful downstream answering. PROBE consistently outperforms both the untrained retriever with inference-time reranking and ReSCORE across all three criteria. These results suggest that final-hop distillation does not restrict the retriever to well-resolved states: the learned relevance preferences remain useful when intermediate queries are incomplete, allowing later retrieval steps to compensate for imperfect early retrieval.

C.4 Query Length and Truncation Analysis

Equation (2) defines the search query before tokenization. In implementation, the concatenated query is encoded with a maximum length of 512 tokens, and queries exceeding this limit are truncated. To examine whether the accumulated summaries and prospective queries lead to substantial truncation, we collect query-length statistics over the same 500 evaluation examples per dataset.

As shown in Table 9, the average retrieval query contains only 67.65 tokens across all three benchmarks, and only 11 of 3,479 queries (0.32%) exceed the 512-token limit. MuSiQue produces longer queries because its questions generally require deeper compositional reasoning, but even there fewer than 1% of queries are truncated. Thus, although Eq. (2) accumulates previous summaries and prospective queries, truncation affects only a small fraction of the evaluated retrieval steps.

C.5 Sensitivity to the Margin Threshold ϵ

PROBE removes training instances whose top-pair Cross-Encoder score margin is below ϵ . The teacher scores are sigmoid-normalized Cross-Encoder logits in $[0, 1]$, whose distribution is strongly concentrated near the endpoints. In particular, the median top-1 and top-2 scores are 0.999612 and 0.996880, respectively. We therefore interpret ϵ as a *near-tie rejection threshold* under sigmoid compression rather than as an absolute

Method	NQ		TriviaQA		SQuAD		AVG	
	EM	F1	EM	F1	EM	F1	EM	F1
<i>Methods without Retriever Training</i>								
SELF-RAG (Llama2-7B+Contriever)	40.8	49.0	37.6	51.6	11.6	25.4	30.0	42.0
IRCoT (Llama-3.1-8B+Contriever)	32.6	41.6	48.8	58.9	24.6	37.2	35.3	45.9
Search-R1 (Qwen2.5-7B+Contriever)	45.0	53.8	57.8	66.6	25.0	38.4	42.6	52.9
<i>Methods with Retriever Training</i>								
ReSCORE (Llama-3.1-8B+Contriever)	41.6	55.4	<u>58.8</u>	69.4	26.2	41.1	42.2	55.3
<i>Our Framework</i>								
PROBE untrained (Llama-3.1-8B+Contriever)	41.4	54.6	59.0	<u>69.9</u>	<u>28.2</u>	<u>42.9</u>	<u>42.9</u>	<u>55.8</u>
PROBE trained (trained Contriever)	<u>42.0</u>	<u>54.7</u>	59.0	70.0	29.4	44.0	43.5	56.2

Table 6: **Additional single-hop QA results on NQ, TriviaQA, and SQuAD.** Bold indicates the best result and underline indicates the second-best.

Reformulation Method	QA		MHR _i @8			Metric	PROBE	Untrained	ReSCORE
	EM	F1	$i = 1$	$i = 2$	$i = \text{final}$				
MuSiQue						Retrieval recovery	62.7	59.2	48.5
LLM-rewrite	16.6	26.0	51.6	61.6	62.7	Full-chain recovery	50.9	46.2	40.0
Thought-concat	21.0	31.0	51.6	63.4	64.5	Semantic answer recovery	37.1	34.9	25.6
PROBE	25.0	34.5	51.6	67.5	68.3				
HotpotQA									
LLM-rewrite	46.2	58.9	68.4	75.6	75.9				
Thought-concat	50.0	62.7	68.4	84.5	85.2				
PROBE	53.4	66.7	68.4	85.8	85.9				
2WikiMHQA									
LLM-rewrite	44.4	53.0	66.3	76.5	79.0				
Thought-concat	49.6	57.5	66.3	85.0	85.3				
PROBE	59.2	67.6	66.3	90.6	90.9				

Table 7: **Effect of query reformulation strategies on MHQA.** LLM-rewrite generates standalone queries at each hop without retaining the initial question. Thought-concat appends prior summaries $\{s_1, \dots, s_{i-1}\}$ to the original question. PROBE retrieves using prospective queries while restricting answer generation to the denoised summary.

confidence threshold.

With the default $\epsilon = 10^{-4}$, the filter removes 2,032 of 14,770 candidate instances (13.76%), retaining 12,738 instances for distillation. Importantly, the retained instances have a median top-pair margin of 0.003218, approximately 32 times the threshold, indicating that most retained pairs are not marginal cases immediately above the cutoff. For each threshold, we reconstruct the filtered training set and retrain the retriever using the same optimization settings. We further evaluate threshold sensitivity on the same 500 examples per dataset.

As shown in Table 10, performance is relatively

Table 8: **Recovery from incomplete first-hop retrieval (%).** Untrained denotes the off-the-shelf Contriever with inference-time Cross-Encoder reranking. PROBE achieves the highest recovery rate under all three criteria.

stable between $\epsilon = 10^{-5}$ and $\epsilon = 10^{-3}$, while $\epsilon = 10^{-4}$ achieves the best average EM and F1 with moderate filtering. Larger thresholds sharply reduce the supervision pool without improving downstream performance: at $\epsilon = 10^{-1}$, only 7.44% of instances remain and average F1 decreases to 54.0. Conversely, reducing the threshold to 10^{-5} retains nearly all candidate instances but slightly degrades average performance. These results suggest that $\epsilon = 10^{-4}$ provides a favorable balance between rejecting ambiguous near-ties and retaining sufficient supervision.

C.6 Controlled Effectiveness–Efficiency Analysis

To examine the effectiveness–efficiency trade-off, we compare closed-book answering, single-pass RAG variants, and iterative retrieval-based reasoning methods under a controlled evaluation setup. All methods are evaluated on the same 500 examples per dataset using the same hardware configuration and single-request serving protocol described in Appendix C.1. For methods marked with * in Table 11, we use their official implementations while

Dataset	# Queries	Avg.	Median	P95	Max	Truncated
MuSiQue	1,300	89.48	59	295.1	645	10 (0.77%)
HotpotQA	1,077	56.64	48	144.0	445	0 (0.00%)
2WikiMultiHopQA	1,102	52.68	42	136.95	587	1 (0.09%)
Overall	3,479	67.65	–	–	645	11 (0.32%)

Table 9: **Query-length statistics for PROBE.** Lengths are measured after tokenizer encoding and before applying the 512-token truncation limit. Only 11 of 3,479 retrieval queries exceed the limit.

ϵ	Retained instances	MuSiQue EM/F1	HotpotQA EM/F1	2WikiMHQA EM/F1	Avg. EM/F1
10^{-1}	1,099 (7.44%)	24.2/34.7	49.2/61.8	56.0/65.4	43.1/54.0
10^{-2}	4,199 (28.43%)	24.0/34.9	52.2/65.4	58.0/66.4	44.7/55.6
10^{-3}	8,775 (59.41%)	23.8/34.3	51.2/64.4	57.6/66.3	44.2/55.0
10^{-4}	12,738 (86.24%)	25.0/34.5	53.4/66.7	59.2/67.6	45.9/56.3
10^{-5}	14,305 (96.85%)	23.6/34.7	54.2/66.8	57.8/65.6	45.2/55.7

Table 10: **Sensitivity to the Cross-Encoder margin threshold ϵ .** We report the number of retained training instances and downstream EM/F1 on the same evaluation subsets.

retaining the same evaluation subsets and hardware environment. All methods, including PROBE, are rerun under this unified serving setup; therefore, the controlled results may differ slightly from those in Table 1.

The closed-book baseline directly prompts the reader to answer without retrieval. The vanilla single-pass RAG baseline retrieves documents once with Contriever using the original question and feeds the retrieved evidence to the reader. The CoT variant applies the same single retrieval step but additionally prompts the reader to perform chain-of-thought reasoning (Wei et al., 2022). The reranking variant applies a Cross-Encoder over the retrieved candidate set before answer generation. HyDE (Gao et al., 2023) first generates a hypothetical answer document and uses it as the retrieval query, while the query-rewrite variant rewrites the original question into a standalone search query before retrieval. These variants test whether stronger query formulation or reranking can compensate for performing only a single retrieval step.

We further compare against a diverse set of iterative RAG methods. IRCot interleaves reasoning and retrieval, FLARE performs forward-looking retrieval based on predicted low-confidence content, and Adaptive-Note maintains intermediate notes to guide subsequent retrieval. Search-R1 trains the LLM via reinforcement learning to issue search actions, while STRIDE explicitly coordinates dependent retrieval and reasoning steps; following its official setup, our STRIDE reproduction uses the Qwen3-8B model trained by the original method.

ReSCORE is the closest retriever-training baseline, using step-wise LLM-derived supervision to fine-tune the retriever. In contrast, PROBE combines prospective query generation with a retriever trained through final-hop Cross-Encoder distillation.

Table 11 shows a clear trade-off between single-pass efficiency and multi-hop effectiveness. Single-pass RAG variants are generally faster, but their QA performance remains substantially below that of strong iterative methods, indicating that a single retrieval step is often insufficient when the required bridging evidence is not directly exposed by the original question.

Among iterative methods, PROBE achieves the best EM and F1 on all three benchmarks while maintaining consistently low latency. On HotpotQA, PROBE reaches 66.6 F1 with 4.59 s latency, compared with 65.2/6.08 s for Search-R1, 59.4/9.25 s for STRIDE, and 59.3/5.06 s for ReSCORE. Similar trends hold on 2WikiMultiHopQA and MuSiQue, where PROBE remains faster than the strong iterative baselines while achieving the highest QA accuracy. Although SELF-RAG has lower latency, its substantially weaker QA performance indicates that this speed comes at a considerable effectiveness cost. Overall, these results show that PROBE provides a favorable effectiveness–efficiency trade-off rather than improving accuracy through additional inference-time computation alone.

We further examine whether the ranking capability of an inference-time Cross-Encoder can be

Method	HotpotQA			2WikiMultiHopQA			MuSiQue		
	EM	F1	Lat.	EM	F1	Lat.	EM	F1	Lat.
<i>Closed-book baseline</i>									
Llama3.1-8B direct answer	16.4	26.0	0.20	11.8	18.6	0.16	2.2	8.7	0.26
<i>Single-pass RAG variants</i>									
Single-pass Contriever RAG	31.2	42.7	1.58	22.6	28.4	0.61	10.8	16.4	0.45
Single-pass Contriever RAG + CoT	37.4	46.7	2.12	32.4	37.4	1.28	13.8	19.6	1.39
Single-pass Contriever + Rerank	37.4	49.5	1.98	31.6	37.4	1.04	11.8	18.6	0.87
HyDE + Single-pass Contriever	37.6	47.6	3.14	21.6	27.8	1.77	13.0	19.3	1.88
Query Rewrite + Single-pass Contriever	31.4	42.7	1.76	20.8	26.6	0.77	11.6	17.7	0.65
<i>Iterative RAG methods</i>									
SELF-RAG	16.2	29.0	1.73	11.8	22.4	0.63	4.6	12.7	0.56
IRCoT	41.2	51.7	6.90	43.4	54.2	3.25	16.8	26.7	2.84
FLARE*	13.0	26.0	5.42	11.8	25.0	4.32	1.4	9.2	3.12
Adaptive-Note*	42.0	54.0	8.74	36.2	45.4	6.37	12.8	20.6	6.02
Search-R1	52.8	65.2	6.08	54.2	62.9	3.13	24.2	33.7	2.98
STRIDE (Qwen3-8B+Contriever)*	49.4	59.4	9.25	57.0	65.6	6.53	24.6	33.6	6.57
ReSCORE*	47.2	59.3	5.06	50.0	59.7	2.59	23.4	32.7	3.00
PROBE	53.3	66.6	4.59	58.6	66.8	2.39	25.4	34.8	2.82

Table 11: **Controlled effectiveness–efficiency comparison.** Lat. denotes average end-to-end latency in seconds. * Results reproduced using the corresponding official implementations.

absorbed into the dense retriever through distillation. Specifically, we compare an inference-time **Rerank-Augmented** pipeline, which combines an off-the-shelf Contriever with a Cross-Encoder reranker, against the **Rerank-Free** PROBE setup, which uses only the distilled retriever during inference. Both configurations are evaluated under the same single-request serving setup.

Configuration	EM	F1	Lat. (s)	Hops
MuSiQue				
Rerank-Augmented	26.8	38.7	4.27	2.71
Rerank-Free (Ours)	25.4	34.8	2.82	2.60
HotpotQA				
Rerank-Augmented	53.8	66.6	5.04	2.18
Rerank-Free (Ours)	53.3	66.6	4.59	2.15
2WikiMultiHopQA				
Rerank-Augmented	56.4	65.7	3.56	2.27
Rerank-Free (Ours)	58.6	66.8	2.39	2.20

Table 12: Performance, end-to-end latency, and average reasoning hops across three datasets. Our distilled retriever (Rerank-Free) achieves competitive accuracy while substantially reducing latency with slightly fewer average reasoning hops.

As shown in Table 12, the rerank-free configuration preserves competitive QA accuracy while reducing latency by approximately 34% on MuSiQue and 33% on 2WikiMultiHopQA. This suggests that

PROBE transfers useful Cross-Encoder ranking preferences into the bi-encoder retriever, avoiding expensive cross-attention reranking during inference.

C.7 Generalization Across LLM Backbones

Table 13 reports end-to-end QA and step-wise retrieval performance across diverse LLM backbones spanning different architectures and scales, including Llama-3.1-8B, Qwen3-8B, and Qwen3-4B. To isolate the impact of external retrieval from internal generation heuristics, we disable the native thinking mode of the Qwen models.

Under this controlled setting, PROBE consistently improves both QA performance and intermediate retrieval recall across model backbones. The gains also tend to increase at later bridging hops, suggesting that the learned relevance signals are not tied to a specific generator and remain useful as the reasoning trajectory becomes more dependent on previously retrieved evidence.

These results indicate that trajectory-level distillation produces retrieval representations that transfer across different generator families and scales without generator-specific retriever adaptation. A natural extension is to investigate whether stronger reasoning models can further improve the quality of the silver trajectories used for retriever distillation.

Backbone	Retriever	QA Metric		MHR@8		
		EM	F1	<i>i</i> = 1	<i>i</i> = 2	<i>i</i> = final
MuSiQue						
Llama-3.1-8B	BM25 (Baseline)	21.2	31.6	44.6	59.3	60.7
	Contriever (Untrained)	20.4	30.9	48.0	60.5	61.5
	Contriever (PROBE)	25.0	34.5	51.6	67.5	68.3
Qwen3-8B	BM25 (Baseline)	19.8	26.5	44.6	57.7	59.5
	Contriever (Untrained)	19.4	26.8	48.0	59.8	60.6
	Contriever (PROBE)	20.6	29.7	51.6	66.2	67.3
Qwen3-4B	BM25 (Baseline)	17.0	23.0	44.6	57.4	58.9
	Contriever (Untrained)	17.8	25.4	48.0	59.6	61.0
	Contriever (PROBE)	20.4	28.5	51.6	65.7	67.4
HotpotQA						
Llama-3.1-8B	BM25 (Baseline)	45.8	58.7	55.9	70.3	71.5
	Contriever (Untrained)	48.0	60.8	64.2	78.3	79.0
	Contriever (PROBE)	53.4	66.7	68.4	85.8	85.9
Qwen3-8B	BM25 (Baseline)	45.8	56.9	55.9	69.7	70.8
	Contriever (Untrained)	48.0	59.7	64.2	78.4	79.2
	Contriever (PROBE)	51.6	64.1	68.4	85.9	86.0
Qwen3-4B	BM25 (Baseline)	44.0	54.7	55.9	70.0	71.3
	Contriever (Untrained)	46.0	57.3	64.2	79.3	79.7
	Contriever (PROBE)	49.2	60.7	68.4	85.2	85.4
2WikiMultiHopQA						
Llama-3.1-8B	BM25 (Baseline)	53.2	60.7	61.8	81.7	82.6
	Contriever (Untrained)	52.2	61.7	60.7	79.5	80.4
	Contriever (PROBE)	59.2	67.6	66.3	90.6	90.9
Qwen3-8B	BM25 (Baseline)	50.0	57.1	61.8	80.3	81.8
	Contriever (Untrained)	47.6	55.3	60.7	79.3	80.7
	Contriever (PROBE)	52.0	59.7	66.3	90.3	91.0
Qwen3-4B	BM25 (Baseline)	43.6	51.0	61.8	81.5	82.4
	Contriever (Untrained)	44.0	52.0	60.7	78.1	79.5
	Contriever (PROBE)	50.8	57.7	66.3	88.8	89.8

Table 13: **Generalization across different LLM backbones on three MHQA benchmarks.** We report Exact Match (EM), F1, and step-wise Multi-Hop Recall (MHR@8). The results demonstrate that our framework (PROBE) consistently improves both end-to-end QA performance and intermediate retrieval recall, regardless of the underlying LLM employed for reasoning.

D Case Study and Failure Analysis

We complement the aggregate evaluation with two case-level analyses: (i) a systematic taxonomy of PROBE’s remaining failures under the effective retrieval context, and (ii) comparative retrieval trajectories that illustrate how different iterative methods behave across representative reasoning patterns.

Failure Analysis. We conduct the failure analysis under the *top-8 effective-context view*, since only the top-8 retrieved documents are provided to the generator during answer generation. This view focuses on the evidence actually visible to the reader rather than all candidates retrieved in a larger retrieval pool. After removing semantically acceptable mismatches, such as name variants, abbreviations, and harmless granularity differences, we identify 606 genuine failures across the three MHQA benchmarks. Table 14 summarizes the resulting taxonomy.

The most frequent failure type is *entity/relation selection error*, accounting for 261/606 failures

(43.1%). In these cases, PROBE retrieves evidence from a relevant entity neighborhood within the effective context, but the reader selects the wrong target entity, relation, or answer slot. For example, the model may choose a related person, organization, location, or film rather than the exact entity required by the question. This suggests that many remaining errors arise from fine-grained constraint tracking and answer selection over closely related evidence.

The next two categories are *temporal/comparison reasoning error* and *relation-chain error*, accounting for 93/606 (15.3%) and 85/606 (14.0%) failures, respectively. Temporal and comparison errors occur when the model retrieves relevant entities but chooses the wrong date, ordering, or comparison side. Relation-chain errors are especially common on 2WikiMHQA, where questions often require following fine-grained family, spouse, or in-law relations. These errors indicate that stronger retrieval alone is insufficient for all remaining cases; relation-level verification and answer-type checking may further improve robustness.

We also observe *retrieval miss / drift / query loop* failures in 98/606 cases (16.2%). This category includes cases where relevant or answer-bearing evidence never enters the top-8 effective context, as well as cases where later hops repeatedly follow an incorrect or unproductive retrieval direction. For example, in a HotpotQA question asking for the movie starring Katrina Bowden and directed by Sean Anders, the gold page *Sex Drive* is retrieved only as a low-ranked candidate and never enters the context used for answer generation. Later hops repeatedly retrieve pages around *Katrina Bowden*, *Sean Anders*, and *Hold Your Breath*. The trajectory eventually introduces the incorrect constraint “2012 supernatural horror film,” shifting retrieval further toward *Hold Your Breath*. The model therefore predicts *Hold Your Breath* instead of the gold answer *Sex Drive*. This illustrates a retrieval-side failure in which the correct evidence is not exposed to the reader and subsequent query reformulation reinforces the wrong retrieval neighborhood.

Comparative Retrieval Trajectories. To further examine how retrieval behavior evolves across hops, we compare PROBE with ReSCORE, IR-CoT, and FLARE on three representative examples covering different reasoning patterns in Table 15.

The cases reveal a consistent distinction in how intermediate information needs are represented. On

Failure Type	HotpotQA	2WikiMHQA	MuSiQue	Total	Overall Ratio
Entity/relation selection error	84	24	153	261	43.1%
Temporal/comparison reasoning error	7	39	47	93	15.3%
Relation-chain error	1	57	27	85	14.0%
Retrieval miss / drift / query loop	28	10	60	98	16.2%
Other failures	16	16	37	69	11.4%
Total genuine failures	136	146	324	606	100.0%

Table 14: Case-level failure analysis of PROBE after removing semantically acceptable mismatches.

HotpotQA, after resolving *TV Choice*, PROBE explicitly queries its missing publication day and retrieves the gold page at rank 4 in the next hop, whereas the alternative trajectories commit to incorrect entities or values and subsequently reinforce them through retrieval. On MuSiQue, PROBE preserves both the Puritan bridge entity and the target region when querying the missing arrival date, retrieving the relevant history page at rank 1. Notably, FLARE retrieves this page at rank 6 but subsequently loses the location binding in its masked query, causing the trajectory to drift again. On 2WikiMultiHopQA, PROBE resolves both film directors before jointly querying their birth years and retrieves all four gold pages in the top four results, while competing methods either fail to convert the resolved directors into new retrieval targets or lose their explicit entity bindings.

Together, the aggregate failure taxonomy and trajectory-level cases show that retrieval drift and query loops remain an important limitation, but they are not the dominant source of PROBE’s remaining errors. Most failures instead arise from fine-grained entity/relation selection and reasoning over relevant or partially relevant evidence. At the same time, the comparative cases illustrate why explicitly representing the unresolved information need can reduce trajectory drift. Future work may therefore combine stronger prospective query generation with uncertainty-aware retrieval, diversity-aware exploration, relation-level verification, answer-type checking, and backtracking mechanisms for correcting earlier retrieval decisions.

Method	Key state / query transition	Retrieval outcome	Final answer
HotpotQA ID: 5abb03325542992ccd8e7eae <i>Entity-to-attribute bridge: George Rainsford → TV Choice → publication day</i> Question: The magazine that nominated George Rainsford for their Best Actor award in 2017 comes out every week on what day of the week?			
PROBE	Hop 1 identifies <i>TV Choice</i> from the George Rainsford page. The prospective query explicitly asks: “What day of the week does the <i>TV Choice Awards</i> magazine come out?”	Retrieves <i>TV Choice</i> at rank 4 in Hop 2. The query preserves the resolved publication and explicitly requests its missing attribute.	Tuesday ✓
ReSCORE	The accumulated thought substitutes <i>TV Week</i> for <i>TV Choice</i> and repeatedly states that the publication day is unknown.	Retrieval remains around George Rainsford, generic Best Actor pages, Laurence Olivier Awards, and Logie Awards for seven hops. The gold <i>TV Choice</i> page is never retrieved.	TV Week / unresolved ✗
IRCoT	The CoT first states that the magazine is the <i>TV Choice Awards</i> . A later CoT sentence prematurely asserts: “The <i>TV Choice Awards</i> is a magazine that comes out every week on Wednesday.” Each sentence becomes the next retrieval query.	<i>TV Choice</i> appears only at rank 19 after Wednesday has already entered the reasoning and retrieval state.	Wednesday ✗
FLARE	Predicted sentences first identify <i>TV Week</i> and later state that the magazine “comes out every week on a Sunday.” The masked queries retain these proposed entities and values.	Retrieval shifts toward Logie Awards and subsequently toward pages such as <i>Every Sunday</i> and <i>Sunday Pants</i> , reinforcing the provisional answer.	Sunday ✗
MuSiQue ID: 3hop2__6989_37759_74563 <i>Three-hop temporal composition: Rainbow Times → New England; Harvard founders → Puritans; arrival date → 1630</i> Question: When did the religious order that founded Harvard College arrive in the region of the U.S. served by the Rainbow Times?			
PROBE	Hop 1 establishes the Massachusetts Bay/Puritan connection. The prospective query asks: “When did the Puritans arrive in the region served by the Rainbow Times?”	Retrieves <i>History of the Puritans in North America</i> at rank 1 in Hop 2. The query retains the bridge group, target region, and missing temporal relation.	1630 ✓
ReSCORE	The first thought prematurely states that Congregationalist ministers arrived in 1629.	Subsequent retrieval remains dominated by Harvard, Boston, and education-history pages rather than shifting to the history of the Puritans’ arrival.	1629 ✗
IRCoT	The CoT identifies the founders as Congregationalists and later describes their arrival as occurring in the “early 17th century,” but never isolates the exact arrival date as the next retrieval target.	The generated descriptive sentences retrieve broad Boston and denominational material; the relevant Puritan history page is not promoted to a useful position.	Early 17th century ✗
FLARE	A predicted sentence retrieves the Puritan history page at rank 6. However, the next masked query—“They arrived in the region of [MASK] in the early 17th century”—retains only the pronoun while removing the location binding.	Retrieval shifts toward broad colonization and Huguenot pages. Later generation conflates the Puritan arrival with the 1620 Mayflower event and the Massachusetts Bay Colony in 1629.	1620 / 1629 ✗
2WikiMultiHopQA ID: 010d6cab08c711ebbd91ac1f6bf848b6 <i>Two-branch comparison: A Romance of the Air → Harry Revier → 1890; How Moscha Came Back → Phillips Smalley → 1865</i> Question: Which film has the director who was born earlier, A Romance of the Air or How Moscha Came Back?			
PROBE	Hop 1 retrieves both film pages and resolves the directors as Harry Revier and Phillips Smalley. The prospective query asks: “What is the birth year of Harry Revier and Phillips Smalley?”	Hop 2 retrieves all four gold pages in its top four results, including both director biographies. The query binds both branches to the same missing attribute.	How Moscha Came Back ✓
ReSCORE	The thought repeatedly states that the directors’ birth dates are unavailable without converting the resolved directors into new retrieval targets.	Retrieval repeatedly returns the same film pages and other titles containing <i>romance</i> , <i>air</i> , or <i>came back</i> . Neither director biography is retrieved across seven hops.	Unresolved ✗
IRCoT	The first derived sentence incorrectly states: “Ivan Reitman was born in 1946.” This sentence becomes the next retrieval query.	Later retrieval drifts through Jason Reitman, Greg Reitman, Leigh Jason, Paul Ivano, and other unrelated people.	Unrelated birth years ✗
FLARE	The predicted sentence correctly identifies the need to find the directors’ birth dates. However, the masked query refers only to “the directors of the two films” and omits Harry Revier and Phillips Smalley.	Retrieval returns generic or unrelated director pages, including D. W. Griffith, Fred LeRoy Granville, Stanley Kubrick, and Lewis Milestone.	Unable to answer ✗

Table 15: **Qualitative comparison of retrieval trajectories.** Across three representative MHQA examples, PROBE preserves resolved intermediate evidence in prospective queries, while competing methods in these examples exhibit entity substitution, premature commitment, or loss of query bindings that can lead to retrieval drift.

Thought Generation Prompt

THOUGHT_GENERATION_PROMPT = ""

<|start_header_id|>system<|end_header_id|>

You are a next-hop search query generator for a multi-hop QA system.

Your goal is to analyze the current search results and generate the *next search query* to find missing information.

Instructions:

1. **Analyze:** Summarize ONLY the useful facts, including entities and relations, currently present in the documents that are relevant to the question. **Do not mention what is missing in the thought.**

2. **Query:** Based on what you know, generate a **complete, self-contained search query**, preferably a natural language question, to retrieve the information that is still needed.

- **Crucial:** Resolve any pronouns or ambiguity. For example, instead of “Where was *he* born?”, say “Where was *Christopher Nolan* born?”.

Output Format:

You must output a single valid JSON object on one line.

Do NOT output markdown code blocks.

Example:

```
{{"thought": "The documents confirm that the director of 'Inception' is Christopher Nolan.",  
"query": "What is the birth city of Christopher Nolan?"}}
```

<|eot_id|><|start_header_id|>user<|end_header_id|>

Documents:

{documents}

Question:

{question}

Previous Reasoning:

{hints}

Output the JSON object now:

<|eot_id|><|start_header_id|>assistant<|end_header_id|>

""

Figure 4: Prompt used to generate an intermediate reasoning summary and a self-contained next-hop search query for multi-hop question answering.

Answer Generation Prompt

```
ANSWER_GENERATION_PROMPT = ""
```

```
<|start_header_id|>system<|end_header_id|>
```

You will receive three inputs: documents, a question, and hints.
Your task is to answer the given question.

Instructions:

- Carefully read the documents and hints.
- If you know the answer to the question confidently, generate an answer using the provided documents and hints.
- If you do not know the answer, generate "Unknown".

Format:

- Return a JSON object formatted as follows: `{{"answer": "Your Response"}}`
- Your response should be a concise short answer without any explanation.
- Ensure the entire response is on a single line without placeholder variables.

You are a helpful assistant.

```
<|eot_id|><|start_header_id|>user<|end_header_id|>
```

Documents:

```
{documents}
```

Question:

```
{question}
```

Hints:

```
{hints}
```

```
<|eot_id|><|start_header_id|>assistant<|end_header_id|>
```

```
""
```

Figure 5: Prompt used to generate the final short answer from retrieved documents and accumulated reasoning hints.