WHEN DOES RAG NEED STRUCTURE? ENHANCING EVIDENCE RETRIEVAL VIA SELECTIVE STRUCTURAL CALIBRATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieval-augmented generation (RAG) grounds large language model outputs in external knowledge. Graph structures capture relations among candidates, but directly adding structural signals can disrupt a reliable relevance-based ranking. To address this limitation, we propose **TRACER** (**T**ask-Conditioned **R**eadout and **A**nchored **C**alibration for **E**vidence **R**etrieval), a selective structural calibration framework. TRACER merges candidates from embedding and structural retrieval and represents them with simple score-, rank-, source-, and propagation-based features. A lightweight task-conditioned reranker then adjusts a frozen base score through bounded residual and within-query reordering branches while discouraging unnecessary changes. Under our controlled evaluation protocol, TRACER ranks first on every reported retrieval metric and yields the highest average question-answering performance; on 2WikiMultihopQA, its N@5 improves by about 12% relative to the strongest baseline. With candidate access fixed, TRACER exceeds a matched query-adaptive fusion control by 7.96 average N@5 points. Controlled decompositions identify structural evidence and within-query reordering as the primary sources of retrieval improvement.

1 INTRODUCTION

Large language models (LLMs) increasingly rely on retrieval-augmented generation (RAG) to ground their outputs in external knowledge and long-term memory (Du, 2026; Lewis et al., 2020). Most retrievers independently rank memories or passages by their relevance to the query, yielding the *base ranking* (Şakar & Emekci, 2025; Izacard et al., 2022). This classical formulation is effective when evidence is locally recognizable, but becomes unreliable when an answer depends on multiple related pieces of evidence. In long-term interaction, relevant information may be distributed across sessions or mixed with outdated memories (Wu et al., 2024); in multi-hop question answering, intermediate evidence may be weakly related to the original query (Trivedi et al., 2022). Therefore, the retrieval target is not merely a list of individually relevant candidates, but a coherent evidence set.

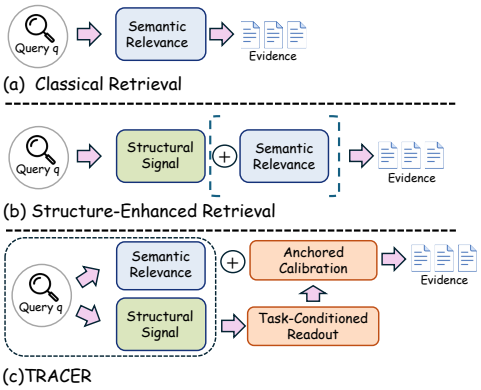

Figure 1: **Retrieval paradigms.** TRACER uses task-conditioned readout and anchored calibration to make candidate-specific structural corrections to a semantic ranking.

Recently, *structure*-enhanced approaches have been introduced to improve RAG by explicitly modeling relationships among retrieval units. Specifically, they organize entities, evidence, or query subproblems into relational structures, such as graphs, hypergraphs, evidence sets, or dependency trees, allowing retrieval to account for their relationships (Jia et al., 2025; Shi et al., 2026). One line of work organizes the corpus itself into relational indexes within knowledge graph, including HippoRAG and GFM-RAG (Gutiérrez et al., 2024; Luo et al., 2026). A complementary line structures

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the retrieval decision around the requirements of the query (Lee et al., 2025). For instance, LogicRAG decomposes queries into subproblems and builds a reasoning graph (Chen et al., 2026b). Related query-adaptive methods estimate retrieval quality or select a retrieval strategy for each query (Meng et al., 2023; Jeong et al., 2024). These approaches adapt retrieval at the query level, whereas TRACER holds the merged candidate pool fixed and learns a separate structural correction for each candidate.

Despite advances, existing structure-enhanced methods primarily focus on how to construct and exploit structures, implicitly assuming that stronger structural signals exert greater influence on retrieval (Luo et al., 2026; Chen et al., 2026a). This assumption is problematic because structural association does not necessarily imply query relevance (Huang et al., 2026). A highly activated candidate may be a necessary bridge, but it may also introduce redundant or outdated evidence (Wu et al., 2024). Without controlling how strongly structure influences individual candidates, structural enhancement can override an already reliable base ranking and make retrieval overly sensitive to dataset-specific structure.

To address this limitation, we propose **TRACER** (Task-Conditioned Readout and Anchored Calibration for Evidence Retrieval), a **selective structural calibration** framework that uses task and candidate context to modulate how structural evidence revises the base ranking, as shown in Figure 1. Here, *selective* denotes task- and candidate-conditioned modulation of structural corrections rather than an explicit binary decision about whether structure is used. Operationally, TRACER performs three steps: merge candidates from the two retrieval paths, summarize them using existing retrieval statistics, and rerank them around the frozen base score. The summaries use compact features such as scores, ranks, margins, source overlap, and graph-propagation values. They require neither a learned graph encoder nor an additional message-passing network. A small task-conditioned readout assigns candidate-specific weights, after which anchored calibration combines a bounded residual with within-query reordering. TRACER therefore reuses the structural backend rather than extending it with another graph model, and its readout is linear in the number of candidates.

Our contributions are threefold:

- We formulate evidence retrieval as a *selective structural calibration* framework, in which task and candidate context produce candidate-specific structural corrections within a fixed candidate union.
- We propose TRACER, a lightweight retrieval model that uses compact retrieval statistics and an asymmetric query–key readout to calibrate a frozen base score without an additional graph encoder.
- Under our controlled evaluation protocol on LongMemEval, 2WikiMultihopQA, and MuSiQue, TRACER ranks first on every reported retrieval metric. It exceeds a matched margin-adaptive fusion control by 7.96 average N@5 points.

2 Related Work

2.1 Relevance-Based Retrieval

Classical retrieval methods estimate query–candidate relevance without explicitly modeling relations among candidates (Lewis et al., 2020). BM25 relies on lexical term statistics, whereas Contriever uses dense embeddings for similarity-based retrieval (Robertson & Walker, 1994; Izacard et al., 2022). Recent approaches further improve retrieval quality through learned reranking or query-side adaptation. ColBERT preserves fine-grained token-level matching for efficient reranking (Khattab & Zaharia, 2020) while BKRAG adopts a BGE reranker to rescore candidates from the initial candidate sets (Guo et al., 2024). Query augmentation methods improve the retrieval input itself: Query2doc generates pseudo-documents for query expansion (Wang et al., 2023), PBR personalizes query expansion using user history (Zhang et al., 2026), and ThinkQE iteratively updates query expansions through retrieval feedback (Lei et al., 2025). For long-term memory retrieval, the full RMM system improves retrieval through retrospective feedback (Tan et al., 2025). These methods strengthen relevance estimation or adapt the retrieval process, but do not explicitly use the relational configuration of the current candidate set to calibrate the base ranking.

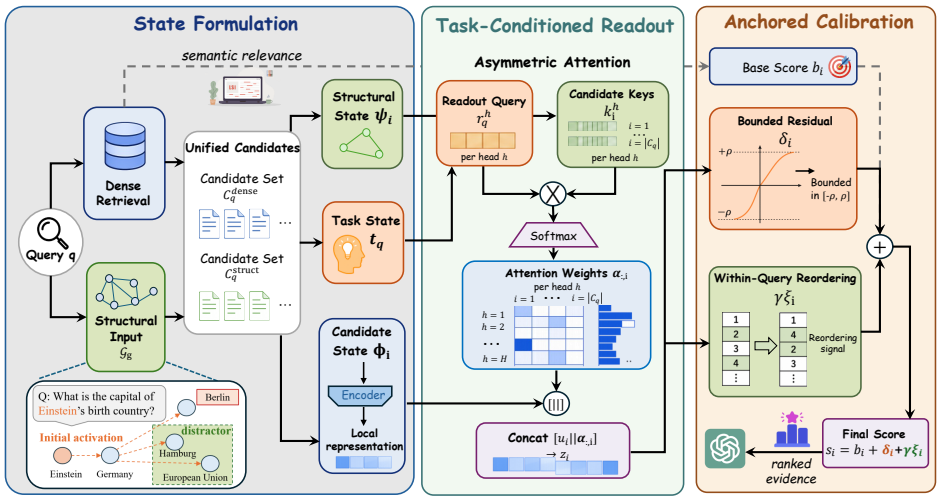

Figure 2: **Overview of the TRACER framework.** TRACER consists of three stages. **State Formulation** merges dense and structural candidates and summarizes existing scores, ranks, source indicators, and propagation statistics. **Task-Conditioned Structural Readout** assigns candidate-specific weights to these summaries. **Base-Score-Anchored Calibration** adjusts the frozen base ranking through bounded residual and within-query reordering.

2.2 Query-Adaptive Retrieval and Fusion

Query-performance prediction estimates retrieval quality without relevance judgments and can guide per-query retrieval decisions (Meng et al., 2023). Adaptive-RAG predicts question complexity to choose among no retrieval, single-step retrieval, and iterative retrieval (Jeong et al., 2024). QuDAR uses top-score margins or LLM judgments to assign query-specific weights to several retrieval lists (Kim et al., 2026). These methods adapt a strategy or a global list weight for each query.

2.3 Structure-Aware Retrieval

Recent methods move beyond independent relevance by introducing structure at the query, candidate-set, or corpus level. SetR identifies the information requirements of a query and selects passages as a collectively sufficient set for retrieval (Lee et al., 2025). Other structured methods retrieve over relational indexes. GraphRAG introduces community-level graph structures for query-focused retrieval, while HippoRAG leverages a knowledge graph and personalized PageRank to support associative retrieval inspired by human memory (Edge et al., 2024; Gutiérrez et al., 2025; 2024). For multi-hop question answering, LogicRAG constructs reasoning graphs, HGRAG models cross-granularity entity–passage relations through hypergraphs, and GFM-RAG performs query-conditioned message passing over knowledge graphs (Chen et al., 2026b; Wang et al., 2026; Luo et al., 2026). These approaches improve how relational evidence is discovered or organized. In contrast, TRACER studies whether the structure observed in a candidate set provides sufficient evidence to revise an otherwise reliable base ranking, and how strongly that revision should be applied.

3 Methodology

3.1 Overview

TRACER is a lightweight reranking layer applied after candidate retrieval. It merges embedding and structural candidates, summarizes fixed retrieval outputs with compact feature vectors, and adjusts a frozen base score through task-conditioned anchored calibration. The model does not learn a graph encoder or perform additional message passing. Its trainable computation is therefore separate from backend-specific graph construction, which is described in Appendix A.2. Algorithm 1 summarizes the shared training and inference procedure.

Algorithm 1 TRACER training and inference

Require: Queries $\mathcal{Q}$; fixed retrieval and structural pipeline; TRACER parameters; training labels when applicable
Ensure: Calibrated rankings and, during training, updated TRACER parameters

```
for all q ∈ Q do
    Construct candidates, frozen base scores, and feature states
    Encode local features and perform task-conditioned structural readout
    Compute anchored calibrated scores and rankings
    if training then
        Mine fixed base-score negatives and evaluate the loss terms
    end if
end for
if training then
    Update TRACER by minimizing the complete training objective
end if
return calibrated rankings
```

3.2 State Formulation

For a query q, fixed embedding and structural backends return two candidate sets. The embedding retriever is equipped with separate low-rank query and candidate score adapters, trained before TRACER and then frozen. The structural backend supplies its native ranking, which we call the structural prior. After deduplication, their union defines the candidates reranked by TRACER:

$$\mathcal{C}_q = \mathcal{C}_q^{\text{dense}} \cup \mathcal{C}_q^{\text{struct}} = \{c_i\}_{i=1}^{N_q}, \qquad b_i = F_{\text{sim}}(q, c_i; \mathcal{C}_q). \tag{1}$$

Here $[N_q] = \{1, \dots, N_q\}$, and the superscript dense denotes the embedding candidate source. The fixed map F_{sim} z-normalizes the frozen adapter scores over the deduplicated set $\mathcal{C}_q$. The resulting b_i values anchor the model and define its base ranking. Let $\mathcal{P}_q \subseteq [N_q]$ be the index set of gold evidence candidates, and let $\mathcal{N}_q = [N_q] \setminus \mathcal{P}_q$ be the negative index set. For the training-query set $\mathcal{Q}_{\text{train}}$, define $\mathcal{Q}_{\text{train}}^{+} = \{q \in \mathcal{Q}_{\text{train}} : \mathcal{P}_q \neq \varnothing\}$.

Let $\mathcal{G}_q$ denote the fixed structural input, such as a topic-level memory structure or an Open Information Extraction (OpenIE) phrase graph (Dong et al., 2022). TRACER consumes compact summaries of the outputs already produced by the retrieval backends. These include scores, ranks, margins, source indicators, overlap counts, PageRank values, and transition summaries rather than learned graph embeddings. Fixed, dataset-specific maps organize these statistics into three feature vectors:

$$\begin{aligned} \boldsymbol{\phi}_i &= F_\phi(q, c_i, \mathcal{C}_q, \mathcal{G}_q), \\ \boldsymbol{\psi}_i &= F_\psi(q, c_i, \mathcal{C}_q, \mathcal{G}_q), \\ \mathbf{t}_q &= F_t(q, \mathcal{C}_q, \mathcal{G}_q). \end{aligned} \tag{2}$$

Under this notation, the uppercase maps F_{sim}, F_ϕ, F_ψ, and F_t are fixed, whereas the lowercase networks f_ϕ and $f_t^{(h)}$ introduced below are trainable. The candidate state $\boldsymbol{\phi}_i \in \mathbb{R}^{d_\phi}$ summarizes local retrieval quality and feeds the candidate-local calibration branch. The structural state $\boldsymbol{\psi}_i \in \mathbb{R}^{d_\psi}$ summarizes the candidate's structural retrieval signals and determines its readout weight. The task state $\mathbf{t}_q \in \mathbb{R}^{d_t}$ summarizes query and candidate-set context shared across candidates. The states are defined by computational role, not statistical independence or dimension-wise semantics; a raw statistic may therefore appear in multiple states.

Before readout, TRACER normalizes the candidate and structural states independently. It also encodes the candidate-local calibration branch:

$$\bar{\boldsymbol{\phi}}_i = \text{LN}_\phi(\boldsymbol{\phi}_i), \qquad \bar{\boldsymbol{\psi}}_i = \text{LN}_\psi(\boldsymbol{\psi}_i), \qquad \mathbf{u}_i = f_\phi(\bar{\boldsymbol{\phi}}_i). \tag{3}$$

where LN_ϕ and LN_ψ are LayerNorm operators (Ba et al., 2016), and f_ϕ is the candidate-state encoder.

3.3 Task-Conditioned Structural Readout

TRACER uses one task query per head instead of comparing every candidate pair. The task state provides the query, and candidate structural states provide the keys. The resulting weights indicate

which available structural summaries should influence each candidate's score. This design avoids both quadratic candidate self-attention and additional graph propagation.

TRACER implements this readout with H heads. For head $h \in \{1, \dots, H\}$, the task state forms a single readout query, and the structural state forms candidate keys:

$$\begin{aligned} \mathbf{r}_q^{(h)} &= f_t^{(h)}(\mathbf{t}_q), \\ \mathbf{k}_i^{(h)} &= \mathbf{W}_k^{(h)} \bar{\boldsymbol{\psi}}_i. \end{aligned} \tag{4}$$

Here $f_t^{(h)}$ and $\mathbf{W}_k^{(h)}$ are trainable head-specific projections. The attention weight measures how well a candidate's structural state matches the task-conditioned readout query (Vaswani et al., 2017). The candidate attention weights are

$$\alpha_{hi} = \mathrm{softmax}_{i \in [N_q]} \left(\frac{\langle \mathbf{r}_q^{(h)}, \mathbf{k}_i^{(h)} \rangle}{\tau \sqrt{d_h}} \right), \tag{5}$$

where $\tau > 0$ is the attention temperature and d_h is the per-head hidden dimension. These weights show how each head distributes attention across candidates under a given task condition. They should not be interpreted as a complete explanation of the final ranking (Jain & Wallace, 2019).

The final score requires a candidate-specific correction. TRACER therefore concatenates the local candidate representation with the candidate's per-head attention weights:

$$\mathbf{v}_i = [\mathbf{u}_i \,\|\, \boldsymbol{\alpha}_{:i}], \tag{6}$$

where $\|$ denotes vector concatenation and $\boldsymbol{\alpha}_{:i} = [\alpha_{1i}, \dots, \alpha_{Hi}]$. This construction keeps the correction candidate-specific while preserving each candidate's contribution to the task-conditioned attention distribution. Unlike candidate self-attention, the readout uses one task-conditioned query per head and does not construct an $N_q \times N_q$ candidate-interaction matrix.

3.4 Base-Score-Anchored Calibration

Once TRACER has encoded the candidate-set structure, it applies structural corrections without replacing the base retriever. This calibration problem has two parts. First, the base score of an individual candidate may be locally overestimated or underestimated. Second, the relative order of candidates within the current set may need to change. TRACER uses separate branches for these two forms of correction.

For any candidate-wise scalar x_i, $\mathrm{z}_q(x_i)$ denotes its population z-score over the candidates in $\mathcal{C}_q$. We add $\epsilon_z = 10^{-6}$ to the standard deviation for numerical stability. The two calibration branches are

$$\begin{aligned} \delta_i &= \rho \tanh(f_\delta(\mathbf{v}_i)), \\ \widetilde{\xi}_i &= \mathrm{z}_q\left(f_\xi(\mathbf{v}_i)\right), \\ \gamma &= \mathrm{sigmoid}(\eta), \end{aligned} \tag{7}$$

The bounded residual branch δ_i operates in the base-score space. Its magnitude is bounded by ρ, so it can correct the retriever score but cannot freely replace it. The within-query reordering branch $\widetilde{\xi}_i$ models how each candidate should move relative to the other candidates in the query. The within-query standardization removes global offset and scale differences across candidate sets while preserving relative reordering directions. The trainable scalar $\gamma = \mathrm{sigmoid}(\eta)$ controls the strength of the within-query reordering branch. It is a single globally learned scalar and does not perform per-query gating. Query-dependent variation in the correction is parameterized by the candidate-conditioned branches f_δ and f_ξ, while the confidence-weighted regularizer $\mathcal{R}_{\mathrm{conf}}$ discourages large departures from a confident base ranking. Together, these components provide candidate-specific, non-uniform calibration without an explicit per-query gate.

The final score is additive around the frozen base score:

$$s_i = b_i + \delta_i + \gamma \widetilde{\xi}_i. \tag{8}$$

The total correction is $\Delta_i = s_i - b_i = \delta_i + \gamma \widetilde{\xi}_i$. f_δ parameterizes the bounded residual branch, f_ξ parameterizes the within-query reordering branch, $\rho > 0$ is the maximum residual magnitude, and η is trainable. This decomposition separates the frozen base score from the bounded residual and within-query reordering branches. The complete update remains anchored through the additive base-score parameterization and the explicit anchor regularizer introduced below.

3.5 Learning Objective

We train TRACER with a listwise evidence loss and pairwise losses over hard, top-ranked, and gain-weighted negatives. Penalties on the bounded residual and total score correction keep the learned ranking near the frozen base score, with stronger protection when the base top-five margin is large. An attention-entropy term stabilizes the structural readout. The complete loss definitions, negative-mining rules, and coefficients are given in Appendix A.5. Together, these terms target difficult ranking inversions while discouraging unsupported departures from a confident base ranking.

3.6 Instantiation and Complexity

TRACER adds a compact reranking stage after fixed embedding and structural backends. Feature assembly only normalizes and groups their existing retrieval statistics; TRACER itself neither constructs a graph nor runs another message-passing model. Its multi-head readout costs $\mathcal{O}(HN_q d_h)$, linear in the candidate count rather than quadratic as in candidate self-attention. The end-to-end system still includes backend-dependent graph construction and propagation. Appendix A reports the complete procedure, and Appendix C provides a CPU query-time decomposition that separates this upstream work from TRACER's added cost.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate TRACER on three benchmarks that cover long-term memory retrieval and multi-hop passage retrieval. **LongMemEval** evaluates retrieval over long-term user–assistant interaction histories, including cases that require multi-session reasoning, temporal reasoning, and knowledge updates (Wu et al., 2024). **2WikiMultihopQA** and **MuSiQue** provide multi-hop questions with supporting evidence paths (Ho et al., 2020; Trivedi et al., 2022). For each dataset, we use 80% for training and 20% for evaluation. We evaluate each setting over five test runs and report the mean. All methods use the same corpus, test split, and metrics while retaining their native candidate-generation procedures. No method uses test labels. We report candidate-pool coverage and reranking results on an identical candidate set to separate candidate access from ranking quality. LongMemEval uses memory chunks derived from interaction histories, whereas 2WikiMultihopQA and MuSiQue use passage chunks.

Baselines. We compare classical BM25, Contriever, and Stella (Robertson & Walker, 1994; Izacard et al., 2022; Zhang et al., 2024); query-augmented PBR and ThinkQE (Zhang et al., 2026; Lei et al., 2025); learned or set-aware BKRAG, a low-rank score adapter following the adapter operation used in RMM, and SetR (Guo et al., 2024; Tan et al., 2025; Lee et al., 2025); and structure-enhanced LogicRAG, HGRAG, GFM-RAG, and HippoRAG-v2 (Chen et al., 2026b; Wang et al., 2026; Luo et al., 2026; Gutiérrez et al., 2025). A matched QuDAR-style control fuses the frozen adapted base ranking with the structural ranking on the same candidate union as TRACER (Kim et al., 2026). We evaluate retrieval with NDCG@k (N@k) and Recall@k (R@k) (Järvelin & Kekäläinen, 2002), and final question-answering performance with Exact Match (EM) and token-level F1 scores.

Implementation details. Contriever provides initial retrieval for TRACER and the reimplemented non-classical baselines. TRACER's base scorer is a dataset-specific low-rank score adapter trained with training-split gold evidence and frozen before TRACER training. Its output defines b_i, the base-score anchor. The *Score adapter* row evaluates this scorer alone; it is not a reproduction of the full RMM memory-management pipeline. Other baselines retain their scoring rules, and BKRAG also uses training-split labels. Thus, Table 1 reports results under our controlled evaluation protocol on a common corpus and data splits. For QA, `GPT-4.1-mini` receives top-5 evidence chunks. Appendix A gives the full feature, fusion, optimization, and hardware settings.

4.2 Main results

Overall retrieval and QA performance. Table 1 compares TRACER with the four retrieval-baseline categories. Under this controlled evaluation protocol, TRACER ranks first on every metric.

Table 1: Retrieval performance across three datasets. The three non-classical baseline categories use Contriever-based results. The **best** and second-best results are highlighted.

Method	LongMemEval				2Wiki				MuSiQue			
	R@3	R@5	N@3	N@5	R@3	R@5	N@3	N@5	R@3	R@5	N@3	N@5
Classical Retrieval												
BM25	64.08	71.50	57.03	57.18	59.75	65.25	67.36	67.89	38.83	44.96	44.16	46.46
Contriever	75.08	79.33	64.38	63.93	52.63	56.50	57.33	57.54	40.54	47.38	46.50	49.13
Stella	68.17	81.75	59.02	62.77	51.88	56.25	59.77	59.83	26.54	31.92	28.54	30.83
Query-Augmented Retrieval												
PBR	76.58	86.58	66.28	68.09	53.50	57.75	57.15	58.00	41.21	48.17	44.74	47.55
ThinkQE	71.58	77.67	61.02	62.67	66.00	74.25	70.99	73.24	48.67	55.88	52.00	54.93
Learned and Set-Aware Retrieval												
BKRAG	73.75	82.92	64.08	65.71	71.38	73.50	80.78	78.83	50.46	56.96	57.69	59.91
Score adapter	80.33	87.75	71.22	72.79	64.00	69.38	71.47	71.71	45.08	51.96	52.03	54.58
SetR	73.67	80.67	61.13	63.06	66.75	67.63	75.81	73.79	52.79	56.38	60.77	61.55
Structure-Enhanced Retrieval												
LogicRAG	75.08	82.00	63.74	64.03	52.63	65.25	57.33	62.47	40.54	51.63	46.50	51.44
HGRAG	72.50	82.33	63.41	65.91	74.25	82.88	79.07	80.99	46.04	52.88	52.01	54.38
GFM-RAG	75.17	83.83	63.11	65.55	73.75	80.63	80.40	81.12	42.54	46.42	45.89	46.71
HippoRAG-v2	73.75	80.58	66.85	67.99	79.63	85.25	85.24	85.21	55.38	62.63	59.82	62.31
TRACER (ours)	**82.08**	**90.25**	**72.18**	**73.70**	**88.88**	**95.38**	**94.79**	**95.58**	**57.54**	**66.13**	**64.88**	**67.81**

Its relative gains over the best baseline score are 1.25–2.85% on LongMemEval, 11.20–12.17% on 2WikiMultihopQA, and 3.90–8.83% on MuSiQue. Table 2 shows the same overall pattern downstream: TRACER leads every dataset-level QA metric and, averaged across datasets, exceeds the best-performing baseline, ThinkQE, by 5.33 EM and 5.80 F1 points.

Table 2: QA performance across three datasets. All retrieval methods use `GPT-4.1-mini` with top-5 evidence, while LLM-only uses raw context. The **best** and second-best results are highlighted.

Method	LongMemEval		2Wiki		MuSiQue		Average	
	EM	F1	EM	F1	EM	F1	EM	F1
LLM-Only								
GPT-4.1-mini	23.00	38.82	22.00	26.81	6.00	14.86	17.00	26.83
Classical Retrieval								
BM25 (Robertson & Walker, 1994)	29.00	36.55	39.50	46.11	19.00	29.56	29.17	37.41
Contriever (Izacard et al., 2022)	30.00	38.49	37.50	44.44	19.00	27.56	28.83	36.83
Stella (Zhang et al., 2024)	32.00	41.69	33.00	39.82	17.00	23.03	27.33	34.85
Query-Augmented Retrieval								
PBR (Zhang et al., 2026)	29.00	39.98	32.00	40.01	25.50	34.72	28.83	38.24
ThinkQE (Lei et al., 2025)	30.00	38.22	49.50	59.71	27.50	38.40	35.67	45.44
Learned and Set-Aware Retrieval								
BKRAG (Guo et al., 2024)	33.00	42.36	48.00	56.87	20.00	32.74	33.67	43.99
Score adapter	32.00	42.48	43.50	52.09	18.00	25.92	31.17	40.16
SetR (Lee et al., 2025)	30.00	39.48	40.50	48.26	26.00	37.06	32.17	41.60
Structure-Enhanced Retrieval								
LogicRAG (Chen et al., 2026b)	25.00	34.83	47.00	54.68	22.00	33.99	31.33	41.17
HGRAG (Wang et al., 2026)	26.00	35.58	51.00	60.43	22.50	31.81	33.17	42.61
GFM-RAG (Luo et al., 2026)	28.00	36.20	55.00	63.60	19.50	27.67	34.17	42.49
HippoRAG-v2 (Gutiérrez et al., 2025)	29.00	36.23	50.50	61.34	27.00	38.20	35.50	45.26
TRACER (ours)	**34.00**	**43.82**	**60.50**	**70.07**	**28.50**	**39.83**	**41.00**	**51.24**

The following controls decompose these gains into candidate access, base-score adaptation, fusion strategy, and component-level reranking.

Table 3: Starting-point decomposition in N@5. Fixed denotes the merged TRACER candidate pool; native denotes Contriever's own pool.

Scorer (pool)	LongMemEval	2Wiki	MuSiQue	Avg.
Contriever (native)	63.93	57.54	49.13	56.87
Contriever (fixed)	62.57	57.54	49.13	56.41
Adapted base scorer (fixed)	70.37	71.71	54.58	65.55

Table 4: Matched fusion on the fixed candidate pool. All methods use the same frozen adapted base scores and structural source scores.

Method	LongMemEval		2Wiki		MuSiQue		Avg. N@5
	R@5	N@5	R@5	N@5	R@5	N@5	
Matched RRF	84.58	66.95	82.12	83.70	58.21	60.57	70.40
Matched equal fusion	85.75	72.18	76.00	80.80	56.83	59.51	70.83
Matched QuDAR-style	86.75	72.58	75.88	80.75	57.46	59.86	71.07
Full TRACER	**90.25**	**73.70**	**95.38**	**95.58**	**66.13**	**67.81**	**79.03**

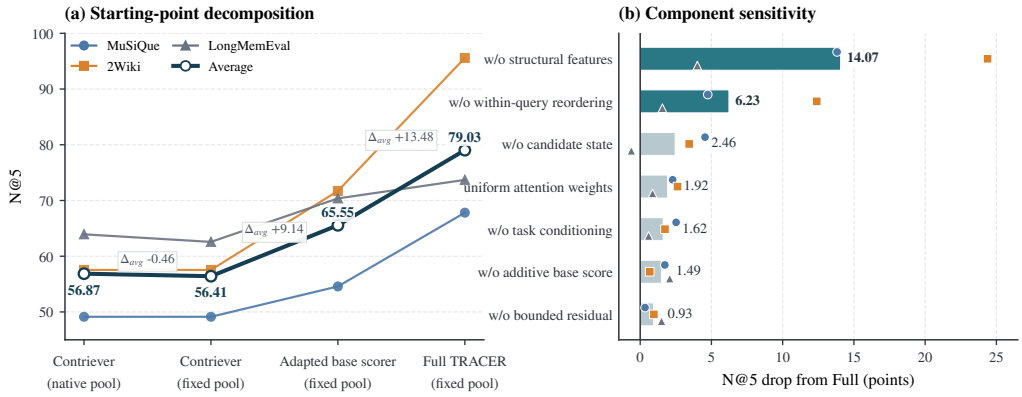

Figure 3: **Decomposition of TRACER's retrieval gains.** (a) N@5 after changing the candidate pool, adapting the base score, and applying TRACER. (b) N@5 change after one-at-a-time ablations on the same candidates.

Candidate access versus ranking on a shared pool. We first measure pre-ranking gold-evidence coverage before attributing the gains to reranking. On 2WikiMultihopQA, the dense top-512 pool covers 83.38% of gold evidence; the structural top-512 pool and its 967.62-candidate union with the dense pool both cover 100.00%. On MuSiQue, dense, structural, and union coverage is 83.04%, 88.67%, and 95.92%, with 966.66 union candidates on average. Holding each union fixed, TRACER improves R@5/N@5 from 69.38/71.71 to 95.38/95.58 on 2WikiMultihopQA and from 51.96/54.58 to 66.13/67.81 on MuSiQue. Thus, the merged pools raise the attainable ceiling, but substantial gains remain when candidate access is held fixed. Table A.5 in Appendix A.6 reports complete test-set pool sizes, ranges, and pre-reranking coverage for all three datasets.

Base-score adaptation and matched fusion. Tables 3 and 4 separate candidate access, base-score adaptation, and reranking. Moving raw Contriever scores to the TRACER pool changes average N@5 from 56.87 to 56.41 (-0.46), whereas the frozen score adapter raises it to 65.55. Among matched controls, QuDAR-style weighting improves average N@5 by only 0.24 points over equal fusion (71.07 versus 70.83). Full TRACER reaches 79.03, gaining 13.48 points over the frozen adapted base scorer and 7.96 over QuDAR-style fusion; the latter gaps are 1.12, 14.83, and 7.95 points on LongMemEval, 2WikiMultihopQA, and MuSiQue. The remaining gain therefore cannot be explained by candidate access, base-score adaptation, or query-level score fusion alone (Figure 3(a)). Appendix B gives the fusion rule and implementation details.

Component attribution. Ablations with candidate identities held fixed localize this gain (Table 5; Figure 3(b)). Withholding structural descriptors from the otherwise identical reranker reduces average N@5 by 14.07 points, while removing within-query reordering costs 6.23 points. The remaining components each contribute at most 2.46 points. Thus, TRACER's main advantage comes from converting structural evidence into candidate-specific ranking corrections, with task-conditioned readout and anchoring providing secondary control.

Table 5: N@5 component ablations on the fixed candidate union. Removing structural features retains graph-retrieved candidates but withholds their structural descriptors.

Variant	LongMemEval	2Wiki	MuSiQue	Avg.
Full TRACER	<u>73.70</u>	**95.58**	**67.81**	**79.03**
w/o structural features	69.69	71.20	53.99	64.96
w/o within-query reordering	72.14	83.19	63.07	72.80
w/o candidate state	**74.32**	92.14	63.26	76.57
Uniform attention weights	72.83	92.96	65.53	77.11
w/o task conditioning	73.11	93.84	65.29	77.41
w/o additive base score	71.63	<u>94.91</u>	66.08	77.54
w/o bounded residual	72.20	94.62	<u>67.47</u>	<u>78.10</u>

Robustness and added cost. Because structural evidence is the largest contributor, Appendix C tests its sensitivity to graph noise. With candidate identities and adapted base scores fixed, 20% structural-signal dropout and misassignment retain 93.50% and 91.58% of clean average N@5, respectively. Severe misassignment instead favors an explicit adapted-base fallback, while TRACER itself adds only 2.8–7.3 ms per query beyond the upstream pipeline.

5 Conclusion

We propose TRACER, a lightweight structure-aware approach that uses compact retrieval statistics to calibrate a frozen base ranking. It merges embedding and structural candidates, applies task-conditioned readout, and adjusts scores through bounded residual and within-query reordering. Under our controlled evaluation protocol, TRACER ranks first on every reported retrieval metric and yields the highest average QA score. A controlled decomposition separates candidate access, base-score adaptation, and reranking. Ablations identify structural features and within-query reordering as the primary contributors, while task-conditioned readout and anchoring provide secondary control over how those corrections are applied. As a limitation, TRACER's gains depend on the availability and quality of upstream structural signals. Controlled corruption shows gradual degradation under moderate noise, but severe structural errors can make the adapted-base fallback preferable.

AI Use Statement

Generative AI tools were used for language editing and limited assistance with code development and debugging. They were also occasionally used to organize or clarify discussions of experimental design and result interpretation. All AI-assisted outputs were independently reviewed and verified by the authors. The authors made all research, methodological, experimental, and interpretive decisions and take full responsibility for the content of this work.

Reproducibility Statement

We describe the problem formulation and all components of TRACER in Sections 3–3.6. Appendix A specifies the notation, dataset-specific feature construction, frozen and trainable components, network and optimization hyperparameters, loss functions, negative-mining rules, and candidate budgets. Appendix B provides the matched shared-pool fusion control, and Appendix C reports structural-signal robustness and query-time cost. Robustness to a different dense retriever and qualitative examples are provided in Appendices D and E, respectively. The datasets, splits, evaluation metrics, and implementation settings used in all experiments are documented in Section 4.1 and Appendix A.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Boyu Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. Pathrag: Pruning graph-based retrieval augmented generation with relational paths. In *Proceedings of the AAAI conference on artificial intelligence*, volume 40, pp. 30183–30191, 2026a.

Shengyuan Chen, Chuang Zhou, Zheng Yuan, Qinggang Zhang, Zeyang Cui, Hao Chen, Yilin Xiao, Jiannong Cao, and Xiao Huang. You don't need pre-built graphs for rag: Retrieval augmented generation with adaptive reasoning structures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30270–30278, 2026b.

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 758–759, 2009. doi: 10.1145/1571941.1572114.

Kuicai Dong, Aixin Sun, Jung-Jae Kim, and Xiaoli Li. Syntactic multi-view learning for open information extraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 4072–4083, 2022.

Pengfei Du. Memory for autonomous llm agents: Mechanisms, evaluation, and emerging frontiers. *arXiv preprint arXiv:2603.07670*, 2026.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Jun Guo, Bojian Chen, Zhichao Zhao, Jindong He, Shichun Chen, Donglan Hu, and Hao Pan. Bkrag: A bge reranker rag for similarity analysis of power project requirements. In *Proceedings of the 2024 6th International Conference on Pattern Recognition and Intelligent Systems*, pp. 14–20, 2024.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models, 2024. NeurIPS 2024.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. From RAG to memory: Non-parametric continual learning for large language models, 2025. ICML 2025.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Manzong Huang, Chenyang Bu, Yi He, Xingrui Zhuo, and Xindong Wu. Relink: Constructing query-driven evidence graph on-the-fly for graphrag. *arXiv preprint arXiv:2601.07192*, 2026.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*, 2022.

Sarthak Jain and Byron C Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3543–3556, 2019.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7036–7050, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.389. URL https://aclanthology.org/2024.naacl-long.389/.

Runsong Jia, Bowen Zhang, Sergio José Rodríguez Méndez, and Pouya G Omran. Structrag: Structure-aware rag framework with scholarly knowledge graph for diverse question answering. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 2567–2573, 2025.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 39–48, 2020.

Joeun Kim, Seunghyouk Yoon, Xuan-Bach Le, Youngeun Nam, Doyoung Kim, Hwanjun Song, and Jae-Gil Lee. QuDAR: Query-wise dual-perspective adaptive retrieval. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 38662–38679, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1791. URL https://aclanthology.org/2026.acl-long.1791/.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Dahyun Lee, Yongrae Jo, Haeju Park, and Moontae Lee. Shifting from ranking to set selection for retrieval augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17606–17619, 2025.

Yibin Lei, Tao Shen, and Andrew Yates. Thinkqe: Query expansion via an evolving thinking process. *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 17772–17781, 2025. doi: 10.18653/v1/2025.findings-emnlp.965.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Linhao Luo, Zicheng Zhao, Reza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. Gfm-rag: graph foundation model for retrieval augmented generation. *Advances in Neural Information Processing Systems*, 38:36371–36405, 2026.

Chuan Meng, Negar Arabzadeh, Mohammad Aliannejadi, and Maarten de Rijke. Query performance prediction: From ad-hoc to conversational search. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2583–2593. Association for Computing Machinery, July 2023. doi: 10.1145/3539618.3591919. URL https://doi.org/10.1145/3539618.3591919.

Stephen E. Robertson and Steve Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 232–241, Dublin, Ireland, 1994. Springer. doi: 10.1007/978-1-4471-2099-5_24.

Tolga Şakar and Hakan Emekci. Maximizing rag efficiency: A comparative analysis of rag methods. *Natural Language Processing*, 31(1):1–25, 2025.

Yuling Shi, Maolin Sun, Zijun Liu, Mo Yang, Yixiong Fang, Tianran Sun, and Xiaodong Gu. Reasoning in trees: Improving retrieval-augmented generation for multi-hop question answering. In *Companion Proceedings of the ACM Web Conference 2026*, pp. 626–635, 2026.

Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long T. Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, Anand Rajan Iyer, Tianlong Chen, Huan Liu, Chen-Yu Lee, and Tomas Pfister. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8416–8439, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.413.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Changjian Wang, Weihong Deng, Weili Guan, Quan Lu, and Ning Jiang. Cross-granularity hypergraph retrieval-augmented generation for multi-hop question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33368–33376, 2026.

Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9414–9423, 2023.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory, 2024.

Dun Zhang, Jiacheng Li, Ziyang Zeng, and Fulong Wang. Jasper and Stella: Distillation of SOTA embedding models, 2024.

Yingyi Zhang, Pengyue Jia, Derong Xu, Yi Wen, Xianneng Li, Yichao Wang, Wenlin Zhang, Xiaopeng Li, Weinan Gan, Huifeng Guo, Yong Liu, and Xiangyu Zhao. Personalize before retrieve: LLM-based personalized query expansion for user-centric retrieval. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16406–16414, 2026. doi: 10.1609/aaai.v40i19.38679.

This appendix provides the detailed notation, dataset-specific instantiations, optimization settings, objective definitions, candidate budgets, retriever robustness analysis, and qualitative examples for TRACER.

A Detailed Implementation

A.1 Notation

Table A.1: Notation. Uppercase $F_\bullet$ maps are fixed; lowercase $f_\bullet$ networks are trainable.

Symbol	Meaning
Inputs and fixed (non-trainable) maps	
$q,\ c_i,\ \mathcal{C}_q$	query, candidate, candidate set
b_i	frozen adapted base score (per-query z-score)
$N_q,\ \mathcal{P}_q,\ \mathcal{N}_q$	candidate count, positive indices, negative indices
F_{sim}	fixed adapted base-score function
$\mathcal{G}_q$	fixed query-specific graph or relational input
$F_\phi,\ F_\psi,\ F_t$	fixed feature-construction maps
$\boldsymbol{\phi}_i,\ \boldsymbol{\psi}_i,\ \mathbf{t}_q$	candidate / structural / task feature vectors (states)
Trainable parameters	
f_ϕ	candidate-state encoder
$f_t^{(h)},\ \mathbf{W}_k^{(h)}$	head-h readout query / key maps
$f_\delta,\ f_\xi,\ \eta$	residual / reorder / global-strength parameters
Derived quantities	
α_{hi}	candidate attention weight (head h, candidate i)
$\delta_i,\ \widetilde{\xi}_i,\ \gamma$	bounded residual, standardized reorder score, global reorder strength
$s_i,\ \Delta_i$	final score, total correction

The state encoders use layer normalization (Ba et al., 2016), and the task-conditioned readout follows the scaled dot-product attention formulation (Vaswani et al., 2017).

A.2 Dataset-Specific Instantiation

The factorization into $\boldsymbol{\phi}_i$, $\boldsymbol{\psi}_i$, and $\mathbf{t}_q$ is shared across LongMemEval (Wu et al., 2024), 2WikiMultihopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022). Only the raw statistics used by the

fixed feature-construction functions depend on the available structural input. Table A.2 summarizes these instantiations.

Table A.2: Dataset-specific instantiations of the shared TRACER feature-vector interface.

Dataset	Candidate unit	Structural input	Candidate construction	Main feature instantiation
LongMemEval	Memory chunk	Topic bank, topic graph, and graph priors	Deduplicated union of retriever output and topic/T3 output	$\boldsymbol{\phi}_i$ contains base-score, successor, rank, and margin features. $\boldsymbol{\psi}_i$ contains successor and graph-prior features. $\mathbf{t}_q$ contains question-type, base-score margin, and source-overlap statistics.
2Wiki	Passage chunk	OpenIE phrase graph with PPR, semantic transitions, and association transitions	Deduplicated union of retriever candidates and graph/PPR candidates	$\boldsymbol{\phi}_i$ contains base-score, similarity, PPR, rank, source, question-type, and candidate-level phrase features. $\boldsymbol{\psi}_i$ contains phrase, propagation, transition, score, rank, and source features. $\mathbf{t}_q$ contains question type, score margins and dispersion, source overlap, and entity-linking statistics.
MuSiQue	Passage chunk	Passage-level OpenIE phrase graph constructed as in 2WikiMultihopQA	The same two-path union as 2WikiMultihopQA	The same feature interface as 2WikiMultihopQA, with dataset-specific data, graph, embedding cache, gold mapping, split, and retriever checkpoint.

Feature-family overlap between $\boldsymbol{\phi}_i$ and $\boldsymbol{\psi}_i$ is intentional. The candidate state feeds the local candidate-quality branch, whereas the structural state defines the key space used by the task-conditioned readout. Specifically, $\boldsymbol{\psi}_i$ is projected into candidate keys and determines candidate attention weights, while $\boldsymbol{\phi}_i$ feeds the candidate-local calibration branch.

A.3 Frozen Components and Trainable Parameters

For LongMemEval, the topic model and adapted embedding-similarity retriever are loaded from the topic-structure pipeline to construct candidates and features. For 2WikiMultihopQA and MuSiQue, the adapted passage-level retriever remains fixed, while OpenIE (Dong et al., 2022), PPR, semantic-transition, and association-transition signals are fixed inputs.

The complete trainable parameter set is

$$\Theta_{\text{TRACER}} = \left\{ \text{LN}_\phi, \text{LN}_\psi, f_\phi, \{f_t^{(h)}, \mathbf{W}_k^{(h)}\}_{h=1}^{H}, f_\delta, f_\xi, \eta \right\}. \tag{A.1}$$

The retriever and all upstream structural pipelines remain frozen during TRACER training. For fixed hidden dimensions, the multi-head readout costs $\mathcal{O}(HN_q d_h)$, which is linear in the candidate count rather than quadratic as in candidate self-attention. The attention weights expose which candidates each head emphasizes under a task condition, while the final score decomposes into the frozen adapted base score, bounded residual, and within-query reordering branches. This pathway makes interventions inspectable relative to the base-score anchor, but attention weights are not treated as a complete explanation.

A.4 Network and Optimization Hyperparameters

Optimization uses AdamW, following the adaptive-moment optimization family introduced by Kingma & Ba (2014).

Here $z_q(x_i) = [x_i - \mu_q(x)]/[\sigma_q(x) + \epsilon_z]$. The mean and population standard deviation are computed over the candidates in $\mathcal{C}_q$. Training instances are shuffled at every epoch. Gradients are accumulated

Table A.3: Network and optimization hyperparameters used by TRACER.

Hyperparameter	Value
Hidden dimension	96
Successor-feature projection dimension	24
Number of readout heads H	4
Attention temperature τ	0.7
Maximum residual ρ	0.9
Direct-score normalization	within-query z-score
Numerical constant ϵ_z	10^{-6}
Optimizer	AdamW
Learning rate	1.5×10^{-3}
Weight decay	1.0×10^{-3}
Gradient accumulation factor	16
Gradient clipping norm	2.0
Maximum epochs, LongMemEval	90
Early-stopping patience, LongMemEval	14
Maximum epochs, 2Wiki/MuSiQue	70
Early-stopping patience, 2Wiki/MuSiQue	10

before each optimizer update. Checkpoints are selected by the validation score

$$\mathrm{R@3} + \mathrm{R@5} + \mathrm{N@3} + \mathrm{N@5}\,. \tag{A.2}$$

A.5 Loss and Negative-Mining Configuration

Ranking-loss definitions. For query q, let $\mathcal{P}_q$ and $\mathcal{N}_q$ denote the positive and negative candidate indices. The listwise loss and the temperature-scaled pairwise loss are

$$\begin{aligned} \mathcal{L}_{\mathrm{list}}(q) &= -\log \frac{\sum_{i\in\mathcal{P}_q} \exp(s_i)}{\sum_{j\in[N_q]} \exp(s_j)}, \\ \ell_{\tau_p}(i,j) &= \log\left[1 + \exp\left(-\frac{s_i - s_j}{\tau_p}\right)\right], \\ \mathcal{L}_{\mathrm{pair}}(q;\mathcal{A},\tau_p) &= \frac{1}{|\mathcal{P}_q|\,|\mathcal{A}|} \sum_{i\in\mathcal{P}_q} \sum_{j\in\mathcal{A}} \ell_{\tau_p}(i,j), \end{aligned} \tag{A.3}$$

Here $\varnothing \neq \mathcal{A} \subseteq \mathcal{N}_q$. Negatives are ranked once by the frozen adapted base score. The K_{hard} highest-ranked negatives form $\mathcal{N}_q^{\mathrm{hard}}$, and their top K_{top} members form $\mathcal{N}_q^{\mathrm{top}}$. The three pairwise terms are

$$\begin{aligned} \mathcal{L}_{\mathrm{hard}}(q) &= \mathcal{L}_{\mathrm{pair}}(q;\mathcal{N}_q^{\mathrm{hard}}, 1.0), \\ \mathcal{L}_{\mathrm{top}}(q) &= \mathcal{L}_{\mathrm{pair}}(q;\mathcal{N}_q^{\mathrm{top}}, 0.7), \\ \mathcal{L}_{\mathrm{gain}}(q) &= \frac{1}{Z_q} \sum_{i\in\mathcal{P}_q} \sum_{k\in\mathcal{N}_q^{\mathrm{hard}}} g_{qik}\,\ell_{0.7}(i,k). \end{aligned} \tag{A.4}$$

Here $g_{qik} \geq 0$ is the supervision weight and $Z_q = \sum_{i\in\mathcal{P}_q} \sum_{k\in\mathcal{N}_q^{\mathrm{hard}}} g_{qik}$. A pairwise term is omitted when its mined negative set is empty, and the gain-weighted term is omitted when $Z_q = 0$.

Regularizer definitions. The regularization terms are

$$\begin{aligned} \mathcal{R}_\delta(q) &= \frac{1}{N_q} \sum_{i=1}^{N_q} |\delta_i|, \\ m_q^{(5)} &= b_{q,(1)} - b_{q,(\min(5,N_q))}, \\ \kappa_q &= \mathrm{clip}\left(\frac{m_q^{(5)} - 1}{2}, 0, 1\right), \end{aligned}$$

Table A.4: Loss coefficients, hard-negative counts, and pairwise temperatures.

Setting	LongMemEval	2WikiMultihopQA	MuSiQue
$\mathcal{L}_{\text{list}}$ coefficient	1.0	1.0	1.0
λ_{hard}	0.08	0.08	0.08
Hard-negative count	12	12	12
Hard-pair temperature	1.0	1.0	1.0
λ_{top}	0.08	0.10	0.10
Top-negative count	3	5	5
Top-pair temperature	0.7	0.7	0.7
λ_{gain}	0.04	0.04	0.04
Gain-pair temperature	0.7	0.7	0.7
λ_{δ}	0.008	0.008	0.008
λ_{conf}	0.02	0.02	0.02
λ_{ent}	0.0005	0.0005	0.0005
λ_{anchor}	0.002	0.002	0.002

$$\begin{aligned}
\mathcal{R}_{\text{conf}}(q) &= \frac{\kappa_q}{N_q}\sum_{i=1}^{N_q}|\Delta_i|,\\
\mathcal{R}_{\text{anchor}}(q) &= \frac{1}{N_q}\sum_{i=1}^{N_q}\Delta_i^2,\\
\mathcal{H}_{\text{attn}}(q) &= -\frac{1}{H}\sum_{h=1}^{H}\sum_{i=1}^{N_q}\alpha_{hi}\log\alpha_{hi},
\end{aligned} \tag{A.5}$$

where $\Delta_i = s_i - b_i$, and $b_{q,(r)}$ is the rth largest retriever z-score for query q, with $b_{qi} \equiv b_i$. If $N_q < 5$, the margin uses the last candidate; clip bounds κ_q to $[0, 1]$, with no additional normalization.

For compactness, define

$$\begin{aligned}
\mathcal{L}_{\text{rank}}(q) &= \mathcal{L}_{\text{list}}(q) + \lambda_{\text{hard}}\mathcal{L}_{\text{hard}}(q)\\
&\quad + \lambda_{\text{top}}\mathcal{L}_{\text{top}}(q) + \lambda_{\text{gain}}\mathcal{L}_{\text{gain}}(q),\\
\mathcal{R}_{\text{cal}}(q) &= \lambda_{\delta}\mathcal{R}_{\delta}(q) + \lambda_{\text{conf}}\mathcal{R}_{\text{conf}}(q)\\
&\quad + \lambda_{\text{anchor}}\mathcal{R}_{\text{anchor}}(q).
\end{aligned}$$

The training objective is

$$\mathcal{L} = \sum_{q\in\mathcal{Q}^{+}_{\text{train}}}\left[\mathcal{L}_{\text{rank}}(q) + \mathcal{R}_{\text{cal}}(q) - \lambda_{\text{ent}}\mathcal{H}_{\text{attn}}(q)\right]. \tag{A.6}$$

All λ terms are non-negative scalar weights; the negative entropy term discourages attention collapse under loss minimization.

Dataset-specific gain-weight construction. All benchmarks use the normalized gain-weighted term in Equation A.4; only the construction of g_{qik} differs.

For LongMemEval, let $\mathcal{Y}_q^{\text{gold}}$ be the set of gold turn identifiers and $\mathcal{T}_{qi}$ the dialogue turn identifiers covered by candidate c_i. Each positive–hard-negative pair is weighted by the positive candidate's gold-turn overlap count:

$$g_{qik} = |\mathcal{T}_{qi} \cap \mathcal{Y}_q^{\text{gold}}|. \tag{A.7}$$

For 2WikiMultihopQA and MuSiQue, every valid positive–hard-negative pair receives $g_{qik} = 1$. Consequently, Z_q normalizes graded supervision in LongMemEval and reduces to the number of valid pairs under binary supervision.

For each query, the entropy contribution is $-\lambda_{\text{ent}}\mathcal{H}_{\text{attn}}(q)$. Because the training objective is minimized, larger attention entropy reduces the loss and discourages all heads from collapsing onto the same candidate.

A.6 Candidate and Structural Budgets

For LongMemEval, the frozen adapted embedding-similarity path returns at most 120 candidates. The structural path is the topic-graph/T3 output, comprising up to 60 topic candidates and 60 internal retriever fallbacks. For 2WikiMultihopQA and MuSiQue, the two inputs are the Contriever top-512 list and the OpenIE-PPR top-512 list, respectively. Each final pool is the deduplicated union of its two inputs.

Table A.5: Test-set candidate pools before reranking. Pool sizes are reported as mean [minimum, maximum]. Gold-evidence coverage is a macro average: we compute the fraction of each query's gold evidence contained in a pool and then average across test queries.

	Pool size: mean [min, max]			Gold-evidence coverage (%)		
Dataset	**Dense**	**Structural**	**Union**	**Dense**	**Structural**	**Union**
LongMemEval	114.96 [107, 119]	85.39 [76, 96]	124.94 [114, 137]	98.00	97.75	98.00
2Wiki	512.00 [512, 512]	512.00 [512, 512]	967.62 [841, 1,011]	83.38	100.00	100.00
MuSiQue	512.00 [512, 512]	512.00 [512, 512]	966.66 [881, 1,018]	83.04	88.67	95.92

For 2WikiMultihopQA and MuSiQue, the active OpenIE subgraph is built from the top 256 phrases. In LongMemEval, `merge_candidate_top_c=20` controls topic merging during topic-bank construction and is not an evidence-candidate budget.

B Matched Query-Adaptive Fusion Control

The matched controls rank the same deduplicated candidate union as TRACER using its frozen adapted base scores and structural prior. For source $r \in \{\text{base}, \text{struct}\}$, let $\widetilde{u}_{ri}$ be its per-query min–max normalized score. Candidates absent from source r receive zero contribution from that source. Following QuDAR-confidence (Kim et al., 2026), we compute

$$m_r = \max\big(\widetilde{u}_{r(1)} - \widetilde{u}_{r(2)}, 0\big), \qquad w_r = \frac{\exp(m_r/\tau)}{\sum_{r'} \exp(m_{r'}/\tau)}, \qquad \tau = 2.0, \tag{B.1}$$

and rank candidates by $\sum_r w_r \widetilde{u}_{ri}$. This two-source variant transfers QuDAR's confidence weighting to the inputs shared with TRACER; the full QuDAR pipeline combines four sparse–dense and original–expanded lists. The RRF and equal-score controls use the same two source lists and have no learned parameters (Cormack et al., 2009). All controls are label-free at test time.

The main text reports the full R@5 and N@5 comparison; this appendix specifies how each matched control is constructed.

C Structural-Signal Robustness and Query-Time Cost

We evaluate sensitivity to imperfect structure without changing candidate access. For every query, candidate identities, gold labels, and frozen adapted base scores remain fixed. Structural-signal dropout replaces a random subset of candidate descriptors with their no-structure counterparts. Structural misassignment instead transfers descriptors from randomly permuted candidates, approximating incorrect graph-to-candidate links. We test 20% and 40% perturbation rates with five seeds (11, 23, 37, 53, and 71). Table C.1 reports N@5 retained relative to the paired clean run, which avoids conflating robustness with dataset-level metric scale.

Moderate perturbations degrade performance gradually, although sensitivity differs across datasets. At 20% dropout and misassignment, TRACER retains 93.50% and 91.58% of clean average N@5. The larger 2WikiMultihopQA drops are consistent with the stronger structural contribution shown by the main-text state/readout ablation. Under 40% misassignment, average retention falls to 82.53%, below both adapted-base fallbacks. Thus, the experiment supports using available structure under moderate corruption, but not trusting arbitrarily poor graphs. When a structural backend is unavailable or fails domain-level validation, the native-pool adapted embedding path provides an explicit graph-free fallback.

Table C.1: N@5 retained under fixed-pool structural-signal perturbations (%). Candidate identities and frozen adapted base scores are unchanged. Perturbation entries report mean ± standard deviation over five seeds. The fixed-pool fallback ranks the union by adapted base score; the native-pool fallback restricts ranking to candidates from the adapted embedding path.

Condition	LongMemEval	2Wiki	MuSiQue	Avg.
Clean structure	100.00	100.00	100.00	100.00
20% signal dropout	97.25±1.56	89.38±1.10	93.89±0.83	93.50
40% signal dropout	95.63±1.43	79.21±0.91	88.32±0.96	87.72
20% signal misassignment	97.32±2.04	86.68±1.17	90.73±1.15	91.58
40% signal misassignment	94.09±1.42	71.57±1.60	81.93±2.05	82.53
Adapted-base fallback (fixed pool)	95.70	75.47	80.50	83.89
Adapted-base fallback (native pool)	95.70	74.93	80.26	83.63

Table C.2: CPU query-time decomposition with cached corpus resources. Upstream covers candidate and structural-signal construction; TRACER total covers feature assembly, neural scoring, and top-five selection. Setup is a one-time loading cost and excludes corpus graph construction.

Dataset	Queries	Setup (s)	Upstream (ms/q)	Features (ms/q)	Scoring (ms/q)	TRACER total (ms/q)
LongMemEval	100	2.34	155.2	1.86	0.92	2.79
2WikiMultihopQA	200	12.32	99.7	4.91	2.36	7.31
MuSiQue	200	29.64	142.8	4.99	2.30	7.33

Table C.2 separates the online cost of the frozen upstream pipeline from the added TRACER computation. The upstream stage includes adapted base scoring, structural propagation, candidate-union construction, and active-structure extraction. Corpus resources are cached, and setup reports their one-time loading cost. All measurements use one CPU thread; top-five selection takes less than 0.05 ms per query and is included in the reported TRACER total.

TRACER contributes less than 7% of the measured online time on every dataset; most latency lies in the frozen upstream pipeline. This result supports the claim that the reranker is lightweight once candidates and structural descriptors are available. It does not make the stronger claim that graph construction is free: one-time corpus graph construction is backend-dependent and is not included in Table C.2. The two tested backends show that TRACER can consume either topic-graph or OpenIE descriptors, but they do not establish transfer to arbitrary vertical domains. In such domains, curated ontologies, citation networks, or domain entity relations can instantiate the same interface, while unreliable structure should be replaced by the native adapted-base fallback.

D Robustness Analysis with a Different Retriever

The main experiments use Contriever as the default dense retriever. To determine whether the downstream gains depend on that choice, we repeat the end-to-end QA evaluation with Stella. Table D.1 reports the resulting performance. LongMemEval contains 100 test queries, while 2WikiMultihopQA and MuSiQue contain 200 test queries each. All scores are reported as percentages.

Table D.1: QA performance with Stella across three datasets. All retrieval methods use `GPT-4.1-mini` with top-5 evidence. The **best** and second-best results are highlighted.

Method	LongMemEval		2Wiki		MuSiQue		Average	
	EM	F1	EM	F1	EM	F1	EM	F1
Classical Retrieval								
Stella (Zhang et al., 2024)	32.00	41.69	33.00	39.82	17.00	23.03	27.33	34.85
Query-Augmented Retrieval								
PBR (Zhang et al., 2026)	31.00	40.90	48.00	57.10	24.50	34.18	34.50	44.06
ThinkQE (Lei et al., 2025)	32.00	40.22	40.00	47.76	26.50	36.12	32.83	41.37
Learned and Set-Aware Retrieval								
Score adapter	**35.00**	42.80	43.50	50.37	24.50	34.00	34.33	42.39
SetR (Lee et al., 2025)	28.00	37.93	40.00	47.92	20.50	31.23	29.50	39.03
Structure-Enhanced Retrieval								
LogicRAG (Chen et al., 2026b)	27.00	36.51	49.00	59.12	20.00	30.18	32.00	41.94
HGRAG (Wang et al., 2026)	26.00	36.76	52.50	63.10	23.00	31.94	33.83	43.93
GFM-RAG (Luo et al., 2026)	27.00	36.11	53.00	62.21	19.00	27.84	33.00	42.05
HippoRAG-v2 (Gutiérrez et al., 2025)	28.00	38.19	48.50	58.19	22.00	32.47	32.83	42.95
TRACER (ours)	34.00	**43.97**	**59.50**	**69.94**	**29.50**	**41.19**	**41.00**	**51.70**

Table D.1 shows that the downstream QA gains persist with Stella. TRACER achieves the highest F1 on all three datasets and the highest EM on 2WikiMultihopQA and MuSiQue. Relative to the runners-up, its EM/F1 margins are 6.50/6.84 points on 2WikiMultihopQA and 3.00/5.07 points on MuSiQue. Relative to Stella, its F1 rises by 2.28, 30.12, and 18.16 points on LongMemEval, 2WikiMultihopQA, and MuSiQue, respectively.

The LongMemEval result is more qualified: the score adapter leads TRACER by one EM point, whereas TRACER leads it by 1.17 F1 points. Across datasets, TRACER has the best average EM and F1, exceeding second-best PBR by 6.50 and 7.64 points. Thus, the Stella evaluation supports retriever robustness overall, but not uniform gains on every metric.

E Qualitative Examples of Query-Dependent Structural Correction

We examine two held-out queries illustrating complementary behaviors: TRACER repairs a ranking missing a structural bridge but preserves an already sufficient ranking. These examples are diagnostic and do not establish an explicit gating mechanism.

E.1 Recovering a Missing Bridge Passage

A 2WikiMultihopQA query asks, "What nationality is Lord Frederick Cavendish-Bentinck's father?" Answering it requires evidence about both Lord Frederick and his father, William Cavendish-Bentinck. The frozen adapted base scorer places the subject passage second but the father's passage eighth, leaving the top-five set incomplete.

The structural prior identifies the missing family-relation bridge. TRACER moves the father's passage to rank two and the subject to rank one, placing both gold passages first. R@5 rises from 0.5 to 1.0 and N@5 from 0.387 to 1.0.

E.2 Preserving an Already Reliable Ranking

A LongMemEval query asks how many weeks elapsed between an exchange-program acceptance and the start of pre-departure orientation. The required dates occur in separate records, but the frozen adapted base scorer already retrieves them in the correct discounted order. Its top-1–top-5 margin is 3.1, with R@5 and N@5 both equal to 1.0.

TRACER leaves the top-five set and order unchanged, so both metrics remain 1.0. Across the two cases, it promotes a missing bridge when needed but preserves an already sufficient ranking.