PonsRAG: A Pons-Inspired RAG Bridging Cognitive Islands for Coordinated Long Narrative Reasoning

Rongchen Zhao[1*,2,†], **Yu Chen**[2,†], **Juyuan Wang**[2], **Zhouting Mo**[4]
Jianxing Yu[3], **Wenqing Chen**[1], **Jingping Liu**[1,‡]
[1]School of Software Engineering, Sun Yat-sen University
[2]School of Future Technology, South China University of Technology
[3]School of Artificial Intelligence, Sun Yat-sen University
[4]TikTok Inc
{edwinzhaorc, cyu94987}@gmail.com, liujp68@mail.sysu.edu.cn

Abstract

Long Narrative Reasoning is an essential capability for processing and reasoning over complex narratives. While retrieval-augmented generation provides a promising framework, existing methods still face two critical challenges: cognitive islanding and cross-layer evidence disconnection. To address these issues, we propose PonsRAG, a coordinated RAG framework inspired by the biological pons. PonsRAG consists of two key components: Triple-Layer Indexing, which organizes documents into a connected knowledge structure to bridge cognitive islands, and Coordinated Reasoning, which retrieves evidence across distinct layers and integrates cross-layer information into a unified context. We evaluate PonsRAG on four long-context narrative benchmarks, and experimental results show that it outperforms the strongest baseline, achieving a 11.56% relative improvement in average accuracy on multi-choice tasks.

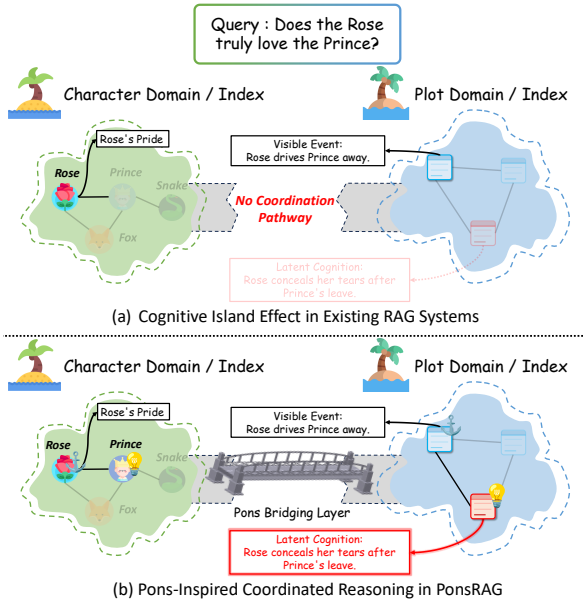


Figure 1: Bridging cognitive islands in long narrative reasoning.

1 Introduction

Long Narrative Reasoning (LNR) refers to the ability of models to process and reason over extended narratives, maintaining context across multiple characters and plots. Unlike multi-hop tasks (Zhou et al., 2025), which connect distant evidence across diverse documents, LNR requires synthesizing information from long and complex texts, making it essential for applications such as advanced dialogue systems, content summarization, and story generation. Retrieval-Augmented Generation (RAG) has emerged as a promising solution for LNR, which builds a retrieval index over the story to retrieve query-relevant evidence for reasoning. Based on their indexing mechanism, recent RAGs can be categorized into two paradigms: Single-Layer and Multi-Layer Indexing. Frameworks adopting **Single-Layer Indexing** organize chunks of the document into a structured knowledge layer as their retrieval index to augment LNR. Much effort has been dedicated to designing various knowledge layers, including HippoRAGv2 (Gutiérrez et al., 2025) and RAPTOR (Sarthi et al., 2024). However, Single-Layer Indexing faces the challenge that relying on a single knowledge layer provides only a partial view of the evidence. For instance, given the query *"Does the Rose truly love the Prince?"* (The Little Prince), a knowledge graph may capture the relationship between the Rose and the Prince but fail to capture higher-level events, such as *the Rose's rejection*, leading to failure. In contrast, **Multi-Layer Indexing** presents a more advantageous solution. It captures distinct views of the evidence by establishing separate knowledge layers. Many studies along this line have demonstrated its effectiveness such as the factual-semantic-episodic index (Wang et al., 2025), the community hierarchical index and the community graph-tree index (Dong et al., 2025). They significantly improve reasoning by providing

*Intern. †Equal contribution. ‡Corresponding author.

multi-view context retrieved from distinct layers. Thus, in this work, we adopt the multi-layer indexing paradigm to support LNR. However, existing methods still face two practical challenges. First, during the offline index construction stage, evidence stored in different knowledge layers is in isolation, making it difficult to connect semantically related information across layers. We use the term Cognitive Island to describe this cross-layer evidence disconnection: character-centric and plot-centric evidence may be individually relevant, yet remain separated during retrieval. For example, the traits and actions of Rose are closely related but may reside in different layers, as illustrated in Figure 1(a). Second, during the online reasoning stage, retrieval is typically conducted independently within each layer, with limited coordination across layers. As a result, the retrieved context may be fragmented, redundant, or even locally biased, making it harder for the model to assemble a coherent reasoning chain. In the example of Figure 1(b), answering the query requires linking evidence across layers, such as *Rose* $\rightarrow$ *Prince* $\rightarrow$ *Rose's rejection* $\rightarrow$ *Rose's tears*. To address these two issues, we propose PonsRAG, a RAG framework inspired by Pons, a neural architecture. It plays an important relay role in coordinating information flow across distributed brain regions (Kandel et al., 2013; Fernández-Gil et al., 2010; Palesi et al., 2017). By linking otherwise separated neural areas through dense fiber pathways, it supports the integration of signals required for complex tasks (Manto et al., 2012; Kratochwil et al., 2017; Zhang et al., 2026b). Based on this theory, PonsRAG introduces a Triple-Layer Indexing architecture with Character, Plot, and Pons layers. The central Pons Layer is implemented as a bipartite graph that connects nodes across distinct layers, enabling cross-layer evidence propagation and selection. Built on this index, PonsRAG further employs a Coordinated Reasoning pipeline that retrieves, matches, and filters evidence jointly across layers, transforming retrieval from isolated layer-wise search into coordinated cross-layer evidence construction. Our primary contributions are summarized as follows:

- We build on the bridge-and-relay concept of the biological pons for LNR, where PonsRAG bridges separated evidence across different knowledge layers, facilitating the integration of cross-layer information.
- We propose PonsRAG, a triple-layer retrieval index paired with a coordinated reasoning pipeline. Its key contribution is shifting from the isolated retrieval of characters and plots to a bridge-constrained joint selection and link fragmented evidence to enable cross-layer reasoning.
- We evaluate PonsRAG on four long-context narrative benchmarks and it achieves the best performance across all benchmarks against all baselines, improving the relative average accuracy by 11.56% on Multi-Choice tasks.

2 Related Work

We classify recent RAGs based on their indexing mechanism into two categories: Single-Layer Index RAGs and Multi-Layer Index RAGs.

2.1 Single-Layer Index RAGs

Existing research into Single-Layer Index RAG typically rely on a monolithic retrieval index to facilitate knowledge acquisition (Chen et al., 2023). For instance, RAPTOR (Sarthi et al., 2024) recursively clusters chunks to build a semantic summary tree which effectively captures events at varying levels of granularity. HippoRAGv2 (Gutiérrez et al., 2025) focuses on relationships between entities by constructing an entity-centric graph over documents and adopts Personalized PageRank (PPR) (Haveliwala, 2002) to retrieve evidence based on the query relative entity. GraphRAG (Microsoft Research, 2024) constructs a knowledge graph index over document-level entities and relations, and augments retrieval by combining local graph evidence with community-level summaries distilled from the graph.

2.2 Multi-Layer Index RAGs

Multi-layered RAG frameworks transcend flat indexing by organizing knowledge into different knowledge layers for retrieval(Zhang et al., 2026a). ComoRAG (Wang et al., 2025) constructs a triple-layer index over veridical, semantic, and episodic knowledge from document, and retrieves evidence from each layer according to a predefined proportion. HiRAG (Huang et al., 2025) constructs multiple knowledge layers from local to global and retrieves evidence by locating high-level relevant contexts and refining to fine-grained evidence. Youtu-GraphRAG (Dong et al., 2025) employs a schema-guided agent to construct a four-layer knowledge tree and retrieves evidence by decomposing com-

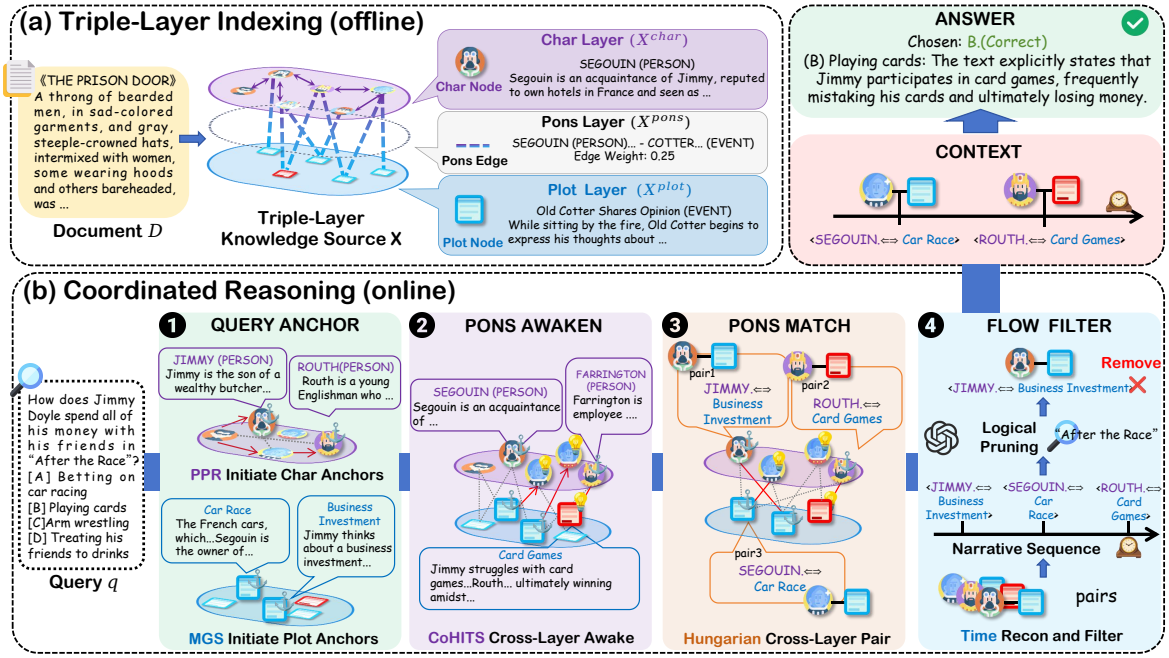


Figure 2: Overall architecture of PonsRAG.

plex queries into schema-aligned sub-queries for retrieval.

3 Overview

In this section, we formalize the problem definition of the long narrative reasoning task and outline our proposed framework to tackle this problem.

3.1 Problem Formulation

Formally, given a long narrative document $\mathcal{D}$ (usually exceeding 200k tokens) and a specific query q, our objective is to generate the optimal answer A. This task is modeled as maximizing the conditional probability:

$$\hat{A} = \underset{A}{\operatorname{argmax}}\, P(A \mid \mathcal{D}, q) \tag{1}$$

3.2 Our Framework

As illustrated in Figure 2, we decouple the framework into two stages:

Triple-Layer Indexing (offline). The framework begins by constructing a knowledge source $\mathcal{X}$ to serve as the index for the document $\mathcal{D}$, viewed from two complementary perspectives. The first layer, Char Layer ($\mathcal{X}^{char}$), models the traits of characters in the narrative context. The second layer, Plot Layer ($\mathcal{X}^{plot}$), captures the narrative progression, consisting of multiple atomic events. To bridge these cognitively isolated layers, we introduce the Pons Layer ($\mathcal{X}^{pons}$), analogous to the biological pons in the neural system. This layer interconnects the character and plot layers, enabling coordinated reasoning across the distributed cognition.

Coordinated Reasoning (online). Building on $\mathcal{X}$, the framework implements a reasoning mechanism that mimics pons-cortex communication. Given a query, coordinated reasoning pipeline obtains the context from $\mathcal{X}$ through four steps: Query Anchor first retrieves the initial matching cognitive nodes from both $\mathcal{X}^{char}$ and $\mathcal{X}^{plot}$, respectively; Pons Awaken then leverages these anchors to discover latent cognitive nodes across layers via $\mathcal{X}^{pons}$; In the Pons Match step, aligned cognition pairs (char, plot) are formed by matching nodes from different layers; Flow Filter reconstructs the chronological plotline from the disordered pairs, eliminates irrelevant noise, and outputs the final plotline as the context. Finally, the framework inputs the final narrative sequence with the query into the Generator to generate the optimal answer $\hat{A}$.

4 Methodology

In this section, we detail the two stages of our framework: Triple-Layer Indexing and Coordinated Reasoning.

4.1 Triple Layer Indexing

Char Layer: Centering Character Traits Given a long narrative document $\mathcal{D}$, we construct a Character Layer $\mathcal{X}^{char}$ from a character-centric

cognition perspective. We first partition $\mathcal{D}$ into a set of chunks, $C = \{c_i\}_{i=1}^{N}$, where N denotes the number of chunks. For each chunk $c_i \in C$, we prompt an LLM to extract entities $\mathcal{E}_i = \{e_{ij}\}_{j=1}^{M_i}$, where M_i denotes the number of extracted entities in chunk c_i, using a pre-defined character-centric schema. To capture the semantic background of each entity, we further instruct the LLM to generate a textual description d_{ij} for each $e_{ij} \in \mathcal{E}_i$. A character node is then defined as $v_{ij}^{char} = (e_{ij}, d_{ij})$, and the set of character nodes extracted from chunk c_i is denoted as $\mathcal{V}_i^{char}$. The complete character node set across all chunks is obtained by $\mathcal{V}^{char} = \bigcup_{i=1}^{N} \mathcal{V}_i^{char}$. To improve retrieval recall, we also instruct the LLM to produce knowledge triples (subject-predicate-object) for each entity e associated with a character node $v^{char} = (e, d_e) \in \mathcal{V}^{char}$, thereby forming a character-centric knowledge graph. These triples are integrated with the character nodes to construct the final Character Layer $\mathcal{X}^{char}$, following a strategy shown to be effective in HippoRAGv2.

Plot Layer: Capturing Plot Progression To model the progression of the narrative, we construct a Plot Layer $\mathcal{X}^{plot}$ based on event-centric cognition along the plotline. For each chunk $c_i \in C$, we record its global position using t_i and use an LLM to extract a set of discrete narrative events $\mathcal{R}_i = \{r_{ij}\}_{j=1}^{K_i}$, where K_i denotes the number of extracted events in chunk c_i. Each event $r_{ij} \in \mathcal{R}_i$ is also assigned an event-type label y_{ij}. To capture high-level context, all extracted events $\mathcal{R} = \bigcup_{i=1}^{N} \mathcal{R}_i$ are then grouped by their labels into clusters $\Phi_y = \{r \in \mathcal{R} \mid y_r = y\}$. For each cluster Φ_y, we use an LLM to generate a global cluster summary $S_y = \text{LLM}_{sum}(\Phi_y)$ and assign this summary to all events within the cluster, i.e., $s_r \leftarrow S_y$ for all $r \in \Phi_y$ Finally, inspired by Zettelkasten(Ahrens, 2017), we encapsulate these multi-granular representations into a Memory Card structure. A plot node is defined as $v^{plot} = (r, y_r, s_r, c_i, t_i)$, where r is an event extracted from chunk c_i at global position t_i. The resulting plot node set $\mathcal{V}^{plot} = \{v^{plot}\}$ constitutes the Plot Layer $\mathcal{X}^{plot}$.

Pons Layer: Bridging Cognitive Islands To establish coordinated reasoning across $\mathcal{X}^{char}$ and $\mathcal{X}^{plot}$, we construct a Pons Layer $\mathcal{X}^{pons}$ that connects character nodes and plot nodes through a weighted Pons edge set $\mathcal{E}^{pons}$, defined as

$$\mathcal{E}^{pons} = \{\, (u, v, w_{uv}) \mid u \in \mathcal{V}^{char},\ v \in \mathcal{V}^{plot} \,\}. \tag{2}$$

The edge weight w_{uv} represents the relevance between a character node u and a plot node v, and integrates two components. (1) Semantic relevance: a normalized semantic similarity $sim(d_u, r_v)$ between the character description d_u and the event r_v, where the sparsity controller τ is defined to remove weak associations; (2) Frequency balancing: an inverse-frequency term $\mathcal{I}_u$, inspired by inverse document frequency (IDF), to mitigate popularity bias favoring high-frequency characters, defined as

$$\mathcal{I}_u = \log\left(\frac{N}{1 + \text{freq}(u)}\right), \tag{3}$$

where N denotes the total number of chunks and $freq(u)$ denotes the number of chunks in which character u appears. The final Pons edge weight w_{uv} is defined as:

$$w_{uv} = \begin{cases} sim(d_u, r_v) \cdot \mathcal{I}_u, & sim(d_u, r_v) \geq \tau \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

4.2 Coordinated Reasoning

Query Anchor: Query-Driven Initialization To initiate coordinated reasoning across the Character Layer $\mathcal{X}^{char}$ and the Plot Layer $\mathcal{X}^{plot}$, we first anchor the query q to a set of seed nodes in both layers. For the Character Layer, we employ Personalized PageRank (PPR) over the character-centric knowledge graph. By performing query-personalized random walks, PPR assigns relevance scores to character nodes $u \in \mathcal{V}^{char}$. The top-k_1 ranked character nodes are selected to form the character anchor set $\mathcal{V}_{anc}^{char}$. For the Plot Layer, we design a Multi-Granularity Scoring (MGS) function to evaluate the relevance between the query q and each plot node $v \in \mathcal{V}^{plot}$. The score jointly considers three complementary views: the narrative event r_v, its corresponding cluster summary s_v, and the original chunk c_v from which the event is extracted. The scoring function is defined as:

$$S(q, v) = \alpha \cdot sim(q, r_v) + \beta \cdot sim(q, s_v) + \gamma \cdot sim(q, c_v) \tag{5}$$

where $sim(\cdot, \cdot)$ denotes cosine similarity, and α, β, γ are hyperparameters controlling the contribution of event-level, summary-level, and chunk-level semantics, respectively. Based on this score,

the top-k_2 plot nodes are selected to form the plot anchor set $\mathcal{V}_{anc}^{plot}$. Consequently, the final anchor set $\mathcal{V}_{anc}$ is defined as the union of anchors from both layers:

$$\mathcal{V}_{anc} = \mathcal{V}_{anc}^{char} \cup \mathcal{V}_{anc}^{plot}. \tag{6}$$

Pons Awaken: Cross-Layer Awakening While the anchor set $\mathcal{V}_{anc}$ provides query-aware surface cognition through semantic retrieval, it remains insufficient for the deeper integration required by coordinated reasoning. Analogous to the biological pons, which serves as a relay and coordination hub for integrating signals across neural pathways, this phase leverages the Pons Layer $\mathcal{X}^{pons}$ to facilitate mutual reinforcement between the Character Layer $\mathcal{X}^{char}$ and the Plot Layer $\mathcal{X}^{plot}$. Specifically, we treat $\mathcal{V}_{anc}$ as query-activated seed signals and propagate relevance from these initial nodes across layers via the Pons connections. To simulate this process, we adopt the Co-HITS ranking algorithm(Deng et al., 2009), which establishes a bidirectional reinforcement loop between character nodes $u \in \mathcal{V}^{char}$ and plot nodes $v \in \mathcal{V}^{plot}$, through the weighted Pons Layer. Formally, this cross-layer resonance process is modeled as:

$$\mathbf{p}^* = \mathcal{H}(\mathcal{V}_{anc}, \mathbf{W}), \mathbf{W}_{uv} = w_{uv}, \tag{7}$$

where $\mathcal{H}(\cdot)$ denotes the Co-HITS propagation operator yielding the steady-state activation $\mathbf{p}^*$, and $\mathbf{W}$ is the weighted adjacency matrix whose entries are given by the Pons edge weights w_{uv} defined in the Pons Layer. Based on the steady-state ranking $\mathbf{p}^*$, we further identify a set of awaken nodes $\mathcal{V}_{awk}$, defined as nodes that achieve high activation scores but are not included in the initial anchor set:

$$\mathcal{V}_{awk} = \{x \mid x \in \text{Top}_{k_3}(\mathbf{p}^*) \wedge x \notin \mathcal{V}_{anc}\} \tag{8}$$

Finally, we construct a candidate subgraph: $\mathcal{G}_{sub} = (\mathcal{V}_{cand}, \mathcal{E}^{sub})$ where $\mathcal{V}_{cand} = \mathcal{V}_{anc} \cup \mathcal{V}_{awk}$ and $\mathcal{E}^{sub} = \{(u, v, w_{uv}) \in \mathcal{E}^{pons} \mid u, v \in \mathcal{V}_{cand}\}$.

Pons Match: Optimal Cross-Layer Pairing After obtaining the candidate subgraph $\mathcal{G}_{sub}$, we further prune it to derive explicit alignment pairs between the Character Layer and the Plot Layer. Given $\mathcal{G}_{sub}$, we formulate the alignment between character nodes $u \in U = \mathcal{V}^{char} \cap \mathcal{V}_{cand}$ and plot nodes $v \in V = \mathcal{V}^{plot} \cap \mathcal{V}_{cand}$ as a Maximum Weight Bipartite Matching problem, where edge weights are defined by the Pons relevance scores w_{uv}. Intuitively, this formulation aims to identify a globally optimal set of character-plot pairs such that each node participates in at most one alignment while the total cross-layer relevance is maximized. To solve this problem, we employ the Hungarian Algorithm (Kuhn, 1955) to compute the optimal matching, retaining the top-k_4 pairs as the final matching set $\mathcal{P}_{match}$, defined as:

$$\mathcal{P}_{match} = \underset{\mathcal{P} \subseteq (U,V)}{\text{argmax}} \sum_{(u,v) \in \mathcal{P}} w_{uv}, \tag{9}$$
$$\forall (u_1, v_1), (u_2, v_2) \in \mathcal{P},\ u_1 \neq u_2 \wedge v_1 \neq v_2.$$

The resulting matching set $\mathcal{P}_{match}$ provides a consistent alignment of the cross-layer interactions activated in the Pons Awaken phase. We restrict Pons Match to a 1-to-1 information bottleneck to isolate the evidence backbone and filter many-to-many narrative noise (see Section 5.4 for 1-to-N relaxations).

Flow Filter: Query-Aware Answer Grounding The matching set $\mathcal{P}_{match}$ consists of unordered character-plot pairs and therefore does not explicitly reflect the temporal progression of the narrative. To recover narrative coherence, we first reorder the matched pairs according to the plot timestamps t_v of their associated plot nodes, producing a chronologically ordered sequence $\mathcal{S}_{match}$. Subsequently, to ensure that the retrieved context satisfies both the semantic intent and the temporal constraints implied by the query q, we introduce an LLM-based filtering module π_{filter}. This module selectively prunes query-irrelevant pairs and resolves temporal references expressed in the query, e.g., "at last" or "earlier", to identify the most relevant subsequence within $\mathcal{S}_{match}$. The final narrative sequence is obtained as $\mathcal{S}_{final} = \pi_{filter}(q, \mathcal{S}_{match})$. Finally, we provide the query q with the filtered narrative sequence $\mathcal{S}_{final}$ as input to the LLM, which generates the final answer A.

5 Experiment

In this section, we present the evaluation results of PonsRAG. We further conduct ablation studies and analytical experiments of our framework.

5.1 Experimental Setup

Benchmarks We conduct experiments on four long context narrative comprehension datasets, spanning both Question Answering (QA) and Multiple Choice (MC) tasks, including NarrativeQA (Kočiský et al., 2018), ∞BENCH (Zhang et al., 2024) (EN.QA and EN.MC) and NoCha (Karpinska et al., 2024) detailed in Appendix A:

Method	NarrativeQA		EN.QA		EN.MC	NoCha	QA Avg.		MC Avg.
	F1	EM	F1	EM	ACC	ACC	F1	EM	ACC
LLM									
GPT-4o-mini	27.29	7.00	29.83	12.82	30.57	60.32	28.56	9.91	45.45
Naive RAG									
BGE-M3(0.3B)	23.16	15.10	23.71	16.24	59.82	56.35	23.44	15.67	58.09
NV-Embed-v2 (7B)	27.18	17.80	34.34	24.57	61.13	68.25	30.76	21.19	64.69
Qwen3-Embed-8B	24.19	15.60	25.79	17.95	65.50	57.14	24.99	16.78	61.32
Structured RAG									
RAPTOR	27.84	17.80	26.33	19.65	57.21	53.17	27.09	18.73	55.19
HippoRAGv2	23.12	15.20	24.45	17.09	60.26	67.46	23.79	16.15	63.86
Youtu-GraphRAG	27.45	15.40	32.03	22.79	68.55	65.87	29.74	19.09	67.21
ComoRAG**(one step)**	29.95	17.60	34.03	24.79	70.31	61.90	31.99	21.20	66.11
PonsRAG (Ours)	**31.19**	**19.00**	**35.13**	**26.21**	**77.73**	**72.22**	**33.16**	**22.61**	**74.98**
Improv.	**+4.14%**	**+6.74%**	**+2.30%**	**+5.73%**	**+10.55%**	**+5.82%**	**+3.66%**	**+6.65 %**	**+11.56%**

Table 1: **Single-step QA performance** on four long narrative comprehension datasets. For fair comparison, we adopt GPT-4o-mini as the LLM backbone. For one-step setting, we limit ComoRAG for max one step. We highlight the **best** and second-best results. **Improv.** denotes the relative improvement of our method over the second-best baseline.

Method	NarrativeQA		EN.QA		EN.MC	NoCha	QA Avg.		MC Avg.
	F1	EM	F1	EM	ACC	ACC	F1	EM	ACC
IRCoT+RAPTOR	31.35	16.00	32.09	19.36	63.76	57.94	31.72	17.68	60.85
IRCoT+HippoRAGv2	28.98	13.00	29.27	18.24	64.19	61.90	29.13	15.62	63.05
IRCoT+Youtu-GraphRAG	25.92	15.40	31.10	22.79	66.38	63.49	28.51	19.10	64.94
ComoRAG**(five steps)**	31.43	18.60	34.52	25.07	72.93	61.90	32.98	21.84	67.42
IRCoT+PonsRAG (Ours)	**33.28**	**20.60**	**36.30**	**26.78**	**79.48**	**72.22**	**34.79**	**23.69**	**75.85**
Improv.	**+5.89%**	**+10.75%**	**+5.16%**	**+6.82%**	**+8.98%**	**+13.75%**	**+5.49%**	**+8.47%**	**+12.51%**

Table 2: **Multi-step QA performance** on four long-narrative comprehension datasets. For fair comparison, we use IRCoT for all enhanced RAGs except ComoRAG, which follows its original multi-step setting with max five steps (details in Section 5.1).

- **NarrativeQA**: A QA dataset comprising books and movie scripts (avg. 58k tokens). Following prior work, we evaluate on a random sample of 500 test questions for computational efficiency.
- **EN.QA**: A QA task from ∞BENCH containing 351 questions on classic novels, with context lengths exceeding 200k tokens.
- **EN.MC**: An MC task from ∞BENCH consisting of 229 questions on classic novels, sharing similar context lengths with EN.QA.
- **NoCha**: An MC dataset comprising 126 True/-False verification questions derived from four public classic novels.

Metrics Following previous work of ComoRAG (Wang et al., 2025) we adopt the F1 score and Exact Match (EM) as evaluation metrics for QA tasks and utilizing Accuracy (ACC) for MC tasks.

Baselines We compare PonsRAG against three baseline categories: **(1) LLM**, which directly process the entire document context (up to 128k tokens). **(2) Naive RAG**, retrieving from flattened 512-token chunks using different embedding models, including BGE-M3 (Chen et al., 2024), NV-Embed-v2 (Lee et al., 2025), and Qwen3-Embed-8B (Zhang et al., 2025). **(3) Structured RAG**, which constructs structure retrieval index, including single-layer index such as RAPTOR and HippoRAGv2, and multi-layer index such as Youtu-GraphRAG and ComoRAG. We separately compare our method against GraphRAG and HiRAG in Appendix E.

Implementation Details For Single-Step QA, all RAGs execute only one single retrieval iteration with GPT-4o-mini as the LLM backbone with context length capped to 6k tokens. For fair compar-

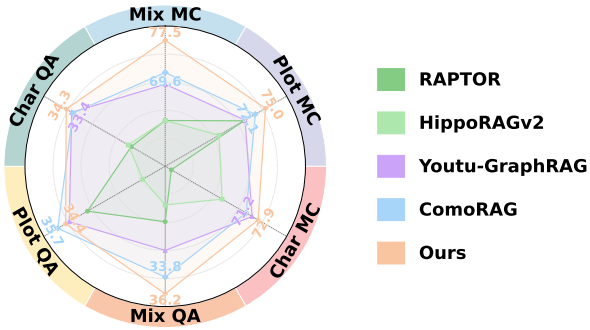


Figure 3: Performance by query types of RAG methods.

Method	EN.MC	EN.QA	
	ACC	F1	EM
PonsRAG	**77.73**	**35.13**	**26.21**
Index			
w/ Char	52.40	21.73	17.95
w/ Plot	55.02	23.37	18.23
w/o Pons	61.13	28.59	19.37
Retrieval			
w/o Pons Awaken	65.50	29.52	24.22
w/o Pons Match	64.19	30.90	21.65
w/o Flow Filter	70.74	32.93	25.07

Table 3: Ablation studies of PonsRAG.

ison, we restrict the Meta Control Loop in ComoRAG to max one round. For Multi-Step QA, we apply IRCoT (Trivedi et al., 2023), which interleaves Chain-of-Thought reasoning with iterative retrieval for those structured RAGs without native multi-step mechanisms. For ComoRAG, we retain its native multi-step setting: a max 5-round Meta Control Loop, which has been proven to be better suited for its index structure than IRCoT in LNR (Wang et al., 2025). For fairness, the total context length across all steps is capped to 6k tokens. We provide the further details about experimental settings and hyperparameters of PonsRAG in Appendix B.

5.2 Main Results

Single-Step QA Performance. From Table 1, we conclude that: 1) Our framework outperforms all baselines across both QA and MC tasks. Notably, on MC tasks, it achieves a relative improvement of 11.56% in average ACC compared to the strongest baseline. This gain demonstrates the robustness and efficacy of our proposed Pons bridging framework in synthesizing fragmented evidence. 2) Remarkably, PonsRAG achieves its largest performance gains on EN.MC, surpassing the second-best method by a margin of over 10%. We attribute this to the longer length documents in EN.MC (150k+ tokens), which exacerbates the cognitive island. Crucially, as this cognitive gap widens, our framework demonstrates an increasing advantage by bridging these isolated islands detailed in Section 5.4.

Multi-Step QA Performance. Table 2 demonstrates that: 1) PonsRAG achieves the highest performance across both QA and MC tasks. Notably, PonsRAG with IRCoT surpasses ComoRAG by over 5% across all benchmarks. 2) The integration of iterative reasoning further unleashes the potential of our mechanisms as the NoCha Improv. increasing from 5.82% to 13.75%. We attribute this boost to the synergy between IRCoT and our architecture. The query rewriting mechanism in IRCoT generates new queries which anchors diverse nodes in Query Anchor stage. With these newly found anchor nodes, Pons Layer uncover hidden nodes that remain dormant under a single static query.

5.3 Ablation Studies

Impact of Triple-layer Knowledge Source. The three rows under Index in Table 3 detail the ablation of our knowledge source (w/ Char, w/ Plot, w/o Pons), revealing several key insights: 1) Relying on either the Char Layer or the Plot Layer yields suboptimal performance, as these single-layer configurations capture only fragmented narrative facts. 2) More importantly, a naive combination of these two layers without the connective Pons Layer remains constrained by the cognitive island problem. The inability to share evidence across these two layers disrupts coordinated reasoning resulting in a performance degradation with ACC dropping by approximately 20% on EN.MC.

Effectiveness of Coordinated Reasoning. To further ablate the coordinated reasoning pipeline, we remove each step except the Query Anchor as it initiates the entire pipeline. The results in Table 3 (three rows under Retrieval) show that each stage is essential, with performance decreasing at varying levels with each removal. Remarkably, the greatest impact is observed when removing Pons Match with ACC dropping by over 10% on EN.MC. We find that this drop is caused by the influence of the main character. In this experiment, we modify the Hungarian Algorithm to select top 30 pairs with the highest edge weight, which leads to a phenomenon where a character matches multiple events. This directly causes the loss of evidence about the character, leading to the performance degradation. We

τ	EN.MC (ACC)	EN.QA (F1)
0.00	72.73 ± 0.51	31.75 ± 0.59
0.25	73.62 ± 0.34	32.58 ± 0.41
0.50	77.31 ± 0.21	$\mathbf{34.61} \pm 0.34$
0.75	$\mathbf{77.60} \pm \mathbf{0.11}$	$34.03 \pm \mathbf{0.20}$
0.80	74.63 ± 0.12	33.12 ± 0.17
0.90	74.20 ± 0.09	32.35 ± 0.13

Table 4: Performance under diverse Sparsity Controller τ (details in Equation (4)).

further conduct experiments about the performance with different matching strategy in Appendix 5.4.

5.4 Detailed Analysis

Longer Document Greater Separation. Inspired by cluster separation theory (Lance and Williams, 1967), we use the average cross-layer semantic distance between character nodes and plot nodes as an indicator of cross-layer separation associated with cognitive island. Specifically, for a character node $u \in \mathcal{V}^{\text{char}}$ and a plot node $v \in \mathcal{V}^{\text{plot}}$, we define their semantic distance as $dis(u, v) = 1 - \cos(d_u, r_v)$, where d_u denotes the description of character u, r_v denotes the event details of the plot node v, and $\cos(\cdot, \cdot)$ denotes cosine similarity. Hence, the average cross-layer distance is then defined as:

$$D_{\text{cross}} = \frac{\sum_{u \in \mathcal{V}^{\text{char}}} \sum_{v \in \mathcal{V}^{\text{plot}}} dis(u, v)}{|\mathcal{V}^{\text{char}}| \cdot |\mathcal{V}^{\text{plot}}|} \tag{10}$$

As illustrated in Figure 4a, D_{cross} rises from 0.231 to 0.377 as document length increases, suggesting that longer documents tend to exhibit greater cross-layer semantic separation between character and plot information.

Greater Separation Greater Gains. To validate the effectiveness of our triple-layer indexing in mitigating cross-layer separation, we compare PonsRAG with a strong baseline under varying levels of D_{cross}. As shown in Figure 4b, the performance gap consistently widens as D_{cross} increases. Specifically, as D_{cross} increases from 0.30 to 0.35, our ACC improves to 70%, whereas ComoRAG drops to 55%, widening the performance gap to 15%. This divergence shows that, under severe cognitive islands, conventional frameworks struggle to connect fragmented evidence stored in distinct layers, whereas the Pons Layer in our index explicitly bridges all layers and turns structural complexity into a retrieval advantage.

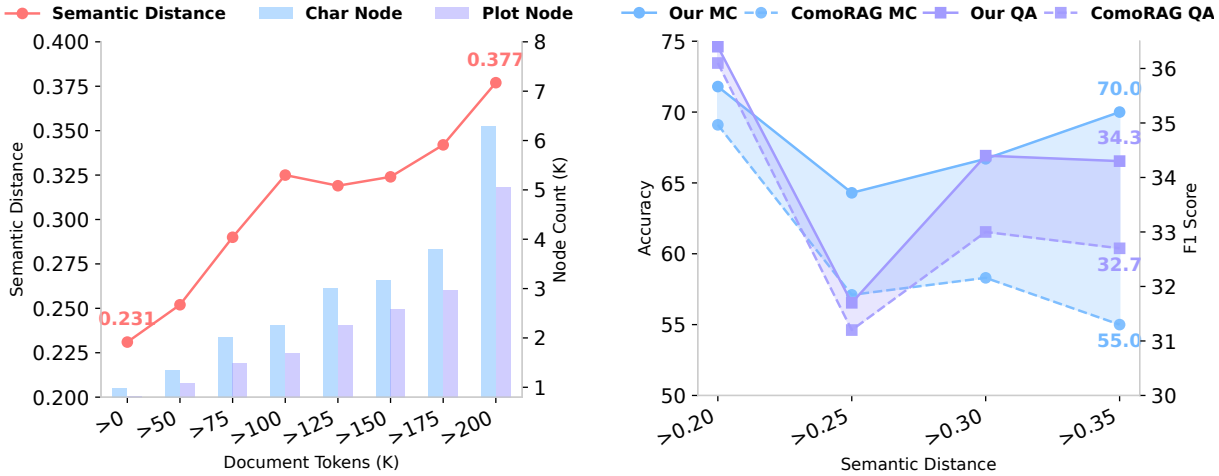
Analysis of Pons Edge. We further analyze the Pons Layer, the core component of our index, with a particular focus on the pons edges it introduces. To determine the key hyperparameter, the Sparsity Controller τ in Equation (4), we examine how performance varies with τ, as shown in Table 4. The results reveal two insights: 1) As τ increases, the variance decreases from 0.51 to 0.09, indicating that performance becomes more stable at higher τ values. 2) Although the best performance is achieved under different settings for different tasks (i.e., $\tau = 0.75$ for MC and $\tau = 0.50$ for QA), performance drops sharply when $\tau < 0.50$ and $\tau > 0.75$. By contrast, when τ is between 0.50 and 0.75, the performance remains steady, with only a 0.58 difference in F1. Thus, the results suggest that the optimal τ lies between 0.50 and 0.75, as this range filters out noisy Pons edges while preserving informative ones. We further study the quality of these pons edges in Appendix C and other vital hyperparameters in Appendix B.3.

Analysis of Query Resolution. To better understand where our method yields the greatest benefit, we categorize all questions from the EN.MC and EN.QA datasets into three types (details about the classification method and statics of each query type are provided in Appendix D):

- **Char Queries**: Queries centering on character traits or background details, e.g., *"What religion is Octavio Amber?"*
- **Plot Queries**: Queries demanding narrative events along the plot line, e.g., *"Where does Trace choose to live at the end of the novel?"*
- **Mix Queries**: Queries necessitating an understanding of both character traits and narrative events, e.g., *"Who is the half crazed man named Arthur who worked with Norbert before Becky?"*

Based on this classification, we compare the performance of PonsRAG and the baseline on each query type. Results in Figure 3 show that the advantage of PonsRAG is most pronounced on Mix queries. Although ComoRAG achieves a 4% lead over PonsRAG on Plot QA, it falls nearly 8% behind on Mix QA. This gap suggests that our coordinated pipeline helps organize character and plot evidence more effectively across layers. To further illustrate the behavior of our coordinated reasoning pipeline, we provide a case study in Appendix F.

Analysis of Matching Constraints Narratives inherently feature many-to-many relationships.



(a) Semantic Distance varying along with document length.

(b) Performance gain along with Semantic Distance.

Figure 4: Analysis of island density and performance gains.

Therefore, we ablate the strict 1-to-1 constraint in Pons Match to test whether a relaxed 1-to-N matching improves reasoning. We compare our approach ($N = 1$) against $N = 2$, $N = 3$, and a Dense setting (retaining all edges in $\mathcal{G}_{sub}$ without pruning). As shown in Table 5, relaxing the mapping

Matching Strategy	EN.MC (ACC)	EN.QA (F1)	Avg Context Tokens
Dense (w/o Match)	68.12	28.65	5,203
1-to-3	73.03	30.34	3,856
1-to-2	74.24	32.28	2,438
1-to-1 (Ours)	**77.73**	**35.13**	**1,187**

Table 5: Ablation on maximum degree constraints in Pons Match.

to 1-to-N ($N \geq 2$) consistently degrades accuracy while substantially inflating the context token load. This confirms that the preceding **Pons Awaken** phase (via Co-HITS) already captures sufficient many-to-many semantic resonance. Consequently, **Pons Match** acts as a crucial sparsity regularizer rather than a recall expander. Routing a denser 1-to-N graph to the **Flow Filter** floods the LLM with cross-layer noise and redundant tokens, severely exacerbating the "lost-in-the-middle" effect.

6 Conclusion

To address the cognitive-island failure mode in long narrative reasoning, we propose PonsRAG, a triple-layer retrieval framework that coordinates Character, Plot, and Pons layers for cross-layer evidence selection. PonsRAG delivers strong performance across four benchmarks, with advantages becoming more evident on longer documents, highlighting the usefulness of structured cross-layer retrieval in long-context settings.

Limitations

Despite its strong performance on long narrative reasoning, PonsRAG still has limitations. Since the framework is explicitly designed for long-context, our evaluation is currently limited to LNR benchmarks. We have not yet examined its effectiveness on other reasoning settings, such as multi-hop QA or more general long-context tasks. Extending the coordinated reasoning paradigm to broader range of reasoning tasks remains an important direction for future work.

Acknowledgments

This paper was supported by the National Natural Science Foundation of China (No. 62306112), Guangdong Basic and Applied Basic Research Foundation (No. 2026A1515010253), and Guangdong S&T Programme Key-Area Research and Development Program of Guangdong Province (2026B0101100004), National Natural Science Foundation of China (62276279), Guangdong Basic and Applied Basic Research Foundation (2024B1515020032).

References

Sönke Ahrens. 2017. *How to Take Smart Notes*. CreateSpace Independent Publishing Platform.

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. *arXiv preprint arXiv:2402.03216*.

Lihan Chen, Tinghui Zhu, Jingping Liu, Jiaqing Liang, and Yanghua Xiao. 2023. End-to-end entity linking

with hierarchical reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4173–4181.

Hongbo Deng, Michael R. Lyu, and Irwin King. 2009. Learning to rank with co-hits. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining (WSDM)*, pages 239–248.

Junnan Dong, Siyu An, Yifei Yu, Qian-Wen Zhang, Linhao Luo, Xiao Huang, Yunsheng Wu, Di Yin, and Xing Sun. 2025. Youtu-GraphRAG: Vertically unified agents for graph retrieval-augmented complex reasoning.

María Ángeles Fernández-Gil, Rosario Palacios-Bote, M Leo-Barahona, and JP Mora-Encinas. 2010. Anatomy of the brainstem: a gaze into the stem of life. *Seminars in Ultrasound, CT and MRI*, 31(3):196–219.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 21497–21515. PMLR.

Taher H. Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th International World Wide Web Conference (WWW)*, pages 517–526.

Haoyu Huang, Yongfeng Huang, Junjie Yang, Zhenyu Pan, Yongqiang Chen, Kaili Ma, Hongzhi Chen, and James Cheng. 2025. Retrieval-augmented generation with hierarchical knowledge. *arXiv preprint arXiv:2503.10150*.

Eric R Kandel, James H Schwartz, Thomas M Jessell, Steven A Siegelbaum, and A James Hudspeth. 2013. *Principles of neural science*, volume 5. McGraw-Hill New York.

Marzena Karpinska, Katherine Thai, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2024. One thousand and one pairs: A "novel" challenge for long-context language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17048–17085. Association for Computational Linguistics.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Claudius F Kratochwil, Upasana Maheshwari, and Filippo M Rijli. 2017. The long journey of pontine nuclei neurons: from rhombic lip to cortico-ponto-cerebellar circuitry. *Frontiers in Neural Circuits*, 11:33.

Harold W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Godfrey N Lance and William Thomas Williams. 1967. A general theory of classificatory sorting strategies: 1. hierarchical systems. *The computer journal*, 9(4):373–380.

P. Langley. 2000. Crafting papers on machine learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pages 1207–1216, Stanford, CA. Morgan Kaufmann.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. NV-Embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*.

Mario Manto, James M Bower, Adriana B Conforto, José M Delgado-García, Sônia N Farias da Guarda, Marcus Gerwig, Christophe Habas, Nobuhiro Hagura, Richard B Ivry, Peter Mariën, et al. 2012. Consensus paper: roles of the cerebellum in motor control—the diversity of ideas on cerebellar involvement in movement. *The Cerebellum*, 11(2):457–487.

Microsoft Research. 2024. Graphrag: Structured retrieval augmented generation. Technical report, Microsoft Research.

Fulvia Palesi, Alessandro De Rinaldis, Gloria Castellazzi, Letizia Casiraghi, Elena Sinforiani, Paolo Vitali, Claudia AM Gandini Wheeler-Kingshott, and Egidio D'Angelo. 2017. Contralateral cortico-ponto-cerebellar pathways reconstruction in humans in vivo: implications for reciprocal cerebro-cerebellar structural connectivity in motor and non-motor areas. *Scientific Reports*, 7(1):12841.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037. Association for Computational Linguistics.

Juyuan Wang, Rongchen Zhao, Wei Wei, Yufeng Wang, Mo Yu, Jie Zhou, Jin Xu, and Liyan Xu. 2025. ComoRAG: A cognitive-inspired memory-organized rag for stateful long narrative reasoning. *arXiv preprint arXiv:2508.10419*.

Mu Zhang, Yuxiang Chu, Guangya Yu, Yongqi Fan, Weiyan Zhang, Hang Hu, Tong Ruan, and Jingping Liu. 2026a. Balancing knowledge breadth and task depth for effective domain adaptation fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 8287–8304.

Ningyu Zhang, Yunzhi Yao, Jiaxin Qin, Haoming Xu, Yuqi Zhu, Zeping Yu, Mengru Wang, Yuqi Tang, Jia-Chen Gu, Shumin Deng, and Huajun Chen. 2026b. Towards principled knowledge editing methods for large language model reasoning. *Nature Machine Intelligence*.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024. ∞Bench: Extending long context evaluation beyond 100K tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15262–15277, Bangkok, Thailand. Association for Computational Linguistics.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Weilin Zhou, Zonghao Ying, Rongchen Zhao, Chunlei Meng, Quanchen Zou, Deyue Zhang, Enhao Gu, Mingze Liu, Dongdong Yang, and Xiangzheng Zhang. 2025. Disentangling fact from sentiment: A dynamic conflict-consensus framework for multimodal fake news detection. *arXiv preprint arXiv:2512.20670*.

A Benchmark Details

In this section, we detail the evaluated benchmarks and summarize their overall statistics as shown in Table 6.

Dataset	#Docs	#Queries	Total Tokens	Avg. Tokens
NarrativeQA	17	500	887,763	52,221
EN.QA	69	351	14,497,149	210,104
EN.MC	58	229	11,249,572	193,958
NoCha	4	126	555,972	138,993

Table 6: Statistics of benchmarks. **Avg. Tokens** denotes the average length of documents.

B Setup and Hyperparameters

B.1 Implementation Details

For fairness, all RAGs use GPT-4o-mini as the backbone with temperature as 0.8 and a 6K-token context limit. All the structured and multi-step RAGs apply BGE-M3 as the embedding model with a 512-token chunk size.

B.2 Hyperparameters Details

For hyperparameters of PonsRAG, we follow HippoRAGv2 to construct the character knowledge graph in Char Layer and we set the sparsity controller τ to 0.75 for MC tasks and 0.50 for QA tasks in Pons Layer. To prevent data leakage, all hyperparameters (including τ and MGS weights) were exclusively optimized on a validation set. More detailed hyperparameters are shown in the Table 7

Hyperparameter	Value
Model Selection	
LLM Backbone Agent	GPT-4o-mini
Retrieval Embedding Model	BGE-M3
Triple-Layer Indexing Setup	
Max Chunk Size	512 tokens
Sparsity Controller (τ)	0.75 (MC) / 0.50 (QA)
Query Anchor Phase	
Anchor Top-K (k_1, k_2)	15
MGS Event Weight (α)	0.7
MGS Summary Weight (β)	0.2
MGS Chunk Weight (γ)	0.1
Pons Awaken & Match Phase	
Awaken Top-K (k_3)	15
Match Top-K (k_4)	30
Context Constraints	
Max Context Length	6,000 tokens

Table 7: Detailed hyperparameter settings of PonsRAG.

B.3 Analysis of MGS Weights

To analyse the MGS weights detailed in Equation (5), we first conduct an ablation study of each weight on EN.MC and EN.QA with results are shown in three rows above the line in Table 8. The results demonstrate that the weight of event r_v contributes most to the performance as it provides the base details about a plot node whereas s_v and c_v can be shared across multiple nodes, potentially introducing noise that degrades performance.

Therefore, we constrain α to be no smaller than β and γ, and use a grid search to identify the best parameter setting. We report representative performance variations under different MGS weight settings in the bottom five rows of Table 8, which suggest two main observations: 1) On both MC and QA tasks, PonsRAG consistently peaks at the setting of $\alpha = 0.7, \beta = 0.2, \gamma = 0.1$. 2) Our performance is relatively insensitive to β and γ weights, with the ACC varying from 74.02 to 77.51, all of which surpass the second-best baseline.

MGS Configuration	EN.MC (ACC)	EN.QA (F1)
$\alpha = 1.0, \beta = 0.0, \gamma = 0.0$	74.02 ± 0.42	33.95 ± 0.55
$\alpha = 0.0, \beta = 1.0, \gamma = 0.0$	69.53 ± 0.48	32.08 ± 0.49
$\alpha = 0.0, \beta = 0.0, \gamma = 1.0$	67.74 ± 0.39	31.15 ± 0.51
$\alpha = 0.5, \beta = 0.1, \gamma = 0.4$	74.25 ± 0.65	33.36 ± 0.71
$\alpha = 0.5, \beta = 0.2, \gamma = 0.3$	74.89 ± 0.48	34.01 ± 0.54
$\alpha = 0.6, \beta = 0.1, \gamma = 0.3$	75.78 ± 0.27	34.67 ± 0.32
$\alpha = 0.6, \beta = 0.2, \gamma = 0.2$	76.22 ± 0.38	34.35 ± 0.48
$\alpha = 0.7, \beta = 0.2, \gamma = 0.1$	$\mathbf{77.51 \pm 0.21}$	$\mathbf{35.13 \pm 0.25}$

Table 8: Performance of PonsRAG with different MGS configurations at $\tau = 0.75$ for EN.MC and $\tau = 0.50$ for EN.QA.

C Detailed Validation of Bridge Quality

To further justify the design of the Pons weight w_{uv} in Equation 4, we compare our method against two common alternative bridging strategies:

- **Entity Mention**: A link is established only if the character's name explicitly appears in the event chunk.
- **Pure Semantic**: Edges are weighted solely by $sim(d_u, s_v)$ without the frequency-balancing term $\mathcal{I}_u$

As shown in Table 9, while Entity Mention achieves high precision, it fails to recover latent narrative links, leading to suboptimal downstream performance. Pure Semantic retrieval suffers from noise introduced by high-frequency characters. Our Pons Weight balances these factors, providing the most effective context for long narrative reasoning.

Bridging Scheme	Prec@100	EN.MC (ACC)	EN.QA (F1)
Entity Mention	91.0%	58.45	24.82
Pure Semantic	64.0%	68.12	30.95
Pons Weight (Ours)	**82.0%**	**75.54**	**34.56**

Table 9: Manual precision of edges and downstream performance on EN.MC and EN.QA across different bridging schemes. Downstream results for Ours are consistent with Table 3.

D Query Type Details

Classification Protocol. To systematically diagnose the reasoning bottlenecks (as discussed in Section 5.4.), we employ GPT-4o as an automated annotator to categorize queries into three distinct types: **Char**, **Plot**, and **Mix**. To quantify the reliability of these automated labels, two human experts independently annotated a random sample of 100 queries. The specific instruction template used for this automated classification is detailed in Appendix H.

Query Distribution. As summarized in Table 10, applying this pipeline to EN.MC and EN.QA yields approximately 30% Char, 26% Plot, and 44% Mix queries. This confirms that the benchmarks heavily emphasize complex joint reasoning while retaining adequate single-aspect coverage.

Query Type	EN.MC	EN.QA	Total	Proportion
Char	59	114	173	29.8%
Plot	68	86	154	26.6%
Mix	102	151	253	43.6%
Total	229	351	580	100.0%

Table 10: Detailed distribution and counts of query types across the EN.MC and EN.QA datasets.

Annotation Reliability. As shown in Table 11, the automated annotator achieved an 88% accuracy against the human consensus, yielding a Cohen's κ of 0.81 (which indicates strong agreement). The confusion matrix reveals that the primary source of discrepancy lies in distinguishing complex Plot queries from Mix queries, since tracking multi-step plot events can sometimes implicitly necessitate character-level reasoning. Nevertheless, the overall misclassification rate remains sufficiently low,

ensuring that our mechanistic observations in Figure 4 are statistically robust.

	Predicted (GPT-4o)			
True (Human)	**Char**	**Plot**	**Mix**	**Total**
Char	25	3	1	29
Plot	1	22	3	26
Mix	0	4	41	45
Accuracy		**88.0%**		

Table 11: Confusion matrix of LLM-based query classification against human annotation on 100 sampled queries.

E Discussion on GraphRAG and HiRAG

In this section, we compare PonsRAG with GraphRAG and HiRAG on a subset of EN.QA. GraphRAG constructs a graph-structured knowledge index and retrieves evidence over entities and their relations. HiRAG adopts a coarse-to-fine retrieval strategy, progressively narrowing from global context to specific details for reasoning. Both methods have shown strong performance on multi-hop reasoning tasks.

However, as shown in Table 12, they struggle with long narrative reasoning, exhibiting both substantially lower performance and much higher cost. In particular, HiRAG achieves only 21.37 (F1), roughly **60%** of our performance, while incurring about **80** $\times$ higher token costs and **8** $\times$ higher time costs. Considering this unfavorable cost–performance trade-off, we exclude them from the remaining benchmarks.

Metrics	**PonsRAG**	**GraphRAG**	**HiRAG**
Performance			
F1 Score	34.38 (100%)	14.60 (42.5%)	21.37 (62.2%)
EM Score	25.31 (100%)	8.20 (32.4%)	14.28 (56.4%)
Token Usage			
Tokens	1.08M (100%)	26.43M (2447%)	95.81M (8871%)
Average Time (s)			
Index	608 (100%)	1936 (318.4%)	4763 (783.4%)
Retrieve	6 (100%)	29 (483.3%)	49 (816.7%)

Table 12: Comparison of performance, token usage, and latency across different RAG paradigms. Percentages indicate the relative ratio compared to PonsRAG.

F Gold Case

To further illustrate the behavior of PonsRAG, Table 13 presents a specific case study. Given an incoming query q: *"How does Jimmy Doyle spend all of his money with his friends in After the Race?"*, existing methods primarily retrieve surface-level event nodes (e.g., *Business Investment*), which often mislead the LLM. In contrast, PonsRAG leverages the character anchor nodes $u \in \mathcal{V}_{anc}^{char}$ (*Jimmy* and *Routh*) initialized during the Query Anchor phase to uncover the latent key evidence $v \in \mathcal{V}_{awk}^{plot}$ (*Card Games*) during the Pons Awaken stage. Subsequently, valid cross-layer connections are formalized into an optimal matching set $\mathcal{P}_{match}$ during the Pons Match phase. Finally, the Flow Filter mechanism prunes distracting noise to reconstruct a coherent chronological storyline $\mathcal{S}_{final}$. This refined context provides the precise evidential support necessary for the Generator to derive the correct answer $\hat{A}$.

Input Data (No Options)

Query: How does Jimmy Doyle spend all of his money with his friends in "After the Race"?
Options: [A] Betting on car racing [B] Playing cards [C] Arm wrestling [D] Treating his friends to drinks

PonsRAG's Choice Result

Query Anchor

Anchor Char Nodes $\mathcal{V}_{anc}^{char}$:
- JIMMY (PERSON): Jimmy is the son of a wealthy butcher, educated in England and known for his popularity and social life...
- ROUTH (PERSON): Routh is a young Englishman who is part of the dinner party...
...

Anchor Plot Nodes $\mathcal{V}_{anc}^{plot}$:
- Car Race: The French cars, which had finished solidly in the race...Segouin is the owner of one of the cars...
- Business Investment: Jimmy thinks about a business investment in the motor industry... believing it to be a good opportunity...
...

Pons Awaken

Awaken Char Nodes $\mathcal{V}_{awk}^{char}$:
- SEGOUIN (PERSON): Segouin is an acquaintance of Jimmy, reputed to own hotels in France...
- FARRINGTON (PERSON): Farrington is an employee who is being reprimanded by Mr Alleyne...
...

Plot Nodes $\mathcal{V}_{awk}^{plot}$:
- Arm Wrestling: Weathers defeats Farrington in a hand wrestling match,...
- Card Games: Jimmy struggles with card games...frequently mistook his cards, leading to confusion and losses...Routh...ultimately winning amidst the excitement of the gathering....
...

Pons Match

Matching Set $\mathcal{P}_{match}$:
- JIMMY... $\Longleftrightarrow$ Business Investment...
- ROUTH... $\Longleftrightarrow$ Card Games...
- FARRINGTON... $\Longleftrightarrow$ Arm Wrestling...
- SEGOUIN... $\Longleftrightarrow$ Car Race...

Flow Filter

Matching Sequence $\mathcal{S}_{match}$:
$\langle$JIMMY, Business Investment$\rangle \Longrightarrow \langle$SEGOUIN, Car Race$\rangle \Longrightarrow$ ... $\Longrightarrow \langle$FARRINGTON, Arm Wrestling$\rangle \Longrightarrow$ ... $\Longrightarrow \langle$ROUTH, Card Games$\rangle \Longrightarrow$ ...
Final Sequence S_{final}:
$\langle$SEGOUIN, Car Race$\rangle \Longrightarrow$ ... $\Longrightarrow \langle$ROUTH, Card Games$\rangle$

Chosen: B.(Correct)
(B) Playing cards: The text explicitly states that Jimmy participates in card games, frequently mistaking his cards and ultimately losing money.

Table 13: **Case Study on Coordinated Narrative Reasoning.** We present a case to demonstrate our model's performance in long-context understanding. Different colors are used to highlight the nature of the processed information: Green is used for key cognition found in step **Query Anchor** that contributes to the correct answer, while Purple is used for the evidence in step **Pons Awaken**.

G Prompting Templates

Below are the detailed instruction templates utilized across the different modules of our framework.

Char Layer **Instruction Template for Entity and Description Extraction**

Role
Your task is to extract entities of specific types from the given text.

Task
For each entity, identify:
1. `entity_name`: capitalized name of the entity
2. `entity_type`: one of the provided types (or "normal_entity" if none match)
3. `entity_description`: a concise description of the entity's attributes and activities

Response Format
Return the result as a list of tuples in the following format:
`("entity"###<entity_name>###<entity_type>###<entity_description>@@@)`
End the response with `<end>`.

Char Layer **Instruction Template for Entity Description Summarization**

Role
You are a helpful assistant responsible for generating a comprehensive summary of the data provided below.

Task
Given one or two entities and a list of related descriptions, synthesize them into a single summary by following these rules:
1. Merge all provided descriptions into a single, comprehensive text, ensuring no collected information is omitted.
2. If the provided descriptions conflict, logically resolve these contradictions to produce a coherent summary.
3. Write strictly in the third person and explicitly include the entity names for full context.

Input Format
```
Entity: ${entity}
Descriptions: ${descriptions}
```

Response Format
Provide a single, comprehensive description that synthesizes all the provided descriptions into a coherent summary.

Plot Layer **Instruction Template for Event Extraction**

Role
You are an expert event extraction assistant.

Task
Please read the following text carefully and extract all key events in chronological order. Make sure:

1. Include all major and minor events mentioned.
2. Maintain chronological order.
3. Each event description should be concise (no more than one sentence).
4. Each detail should clearly explain what happened, who was involved.
5. Do not add any commentary or analysis beyond the events themselves.

Input Format
`Text to analyze: ${chunk_text}`

Response Format
Return the result as a list of tuples in the following format:
`(<event description>###<event details>)@@@`
End the response with `<end>`.

Plot Layer **Instruction Template for Event Summarization**

Role
You are an expert event summarization assistant.

Task
Please read the following events in chronological order carefully and generate a comprehensive summary. Make sure:
1. Include all events in the summary.
2. Include all details about each event.

Input Format
`Events: ${events}`

Response Format
Return the summary directly without any additional commentary or analysis.

Flow Filter **Instruction Template for Character-Event Pair Filtering**

Role
You are a critical component of a high-stakes question-answering system used by top researchers and decision-makers worldwide.

Task
1. Identify pairs that are helpful to answer the user's query, if none are helpful, output `NONE`.
2. Output the specific indices of the selected pairs.

Input
You are given a Question and several Pairs (Characters, Events) from an article.

Response Format
ONLY OUTPUT INDICES SEPARATED BY COMMA!!!
Example: `0,3,5,7,8,10,14`

Limits

- The accuracy of your response is paramount, as it will directly impact the decisions made by these high-level stakeholders!!!
- You must only use character or events from the candidate list and do not generate new ones!!!

QA Answering **Instruction Template for Final Answer Generation**

Role

You are an expert at carefully reading complex texts, extracting narrative details, and making logical inferences.

Task

Given the following detail article from a book, and a related question, you need to provide a comprehensive and accurate answer based on the given information.

Input

The context comprises extracted character profiles and a chronologically ordered sequence of narrative events, supported by their original text chunks.

`Context: {<role_1, event_1> <role_2, event_2>:.<role_n, event_n>}`
`Question: {question}`

Response Format

- **Content Understanding**: Start with a brief summary of the content in no more than two sentences.`### Content Understanding`
- **Relevant Information Analysis**: Provide a markdown list of all relevant evidence strictly from retrieved documents.`### Relevant Information Analysis`
- **Key Facts**: List the key facts that directly support the answer.`### Key Facts`
- **Final Answer**: Provide the shortest possible answer taken directly from the text.`### Final Answer`

Limits

- Do not infer or assume anything not explicitly stated in the retrieved documents.
- Do not fabricate facts.
- Prefer answers supported by multiple independent pieces of evidence.

H Prompt Template for Query Type Classification

Query Classification **Three-shot Demonstration Template**

Three-shot Demonstration:

```
{"QUERY": "What religion is Octavio Amber?"}
{"ID": "char"}

{"QUERY": "Where does Trace choose to live at the end of the novel?"}
{"ID": "plot"}

{"QUERY": "Who is the half crazed man named Arthur who worked with Norbert before
Becky?"}
{"ID": "mix"}
```

Query Classification **Instruction Template**

Role

You are an expert on evaluating and classifying query types.

Task

1. Understand and identify the information needed to answer the given query.
2. Classify the query into three types based on the retrieval focus:
   - `char`, which focuses on character profiles;
   - `plot`, which focuses on narrative events;
   - `mix`, which requires joint reasoning to synthesize relations between character and event.

Input

```
{"QUERY": "XXX"}
```

Output

```
{"ID": "XXX"}
```