Chronological Passage Assembling in RAG framework for Temporal Question Answering

Anonymous ACL submission

Abstract

Long-context question answering over narrative tasks is challenging because correct answers often hinge on reconstructing a coherent timeline of events while preserving contextual flow in a limited context window. Retrieval-augmented generation (RAG) methods aim to address this challenge by selectively retrieving only necessary document segments. However, narrative texts possess unique characteristics that limit the effectiveness of these existing approaches. Specifically, understanding narrative texts requires more than isolated segments, as the broader context and sequential relationships between segments are crucial for comprehension. To address these limitations, we propose ChronoRAG, a novel RAG framework specialized for narrative texts. This approach focuses on two essential aspects: refining dispersed document information into coherent and structured passages and preserving narrative flow by explicitly capturing and maintaining the temporal order among retrieved passages. We empirically demonstrate the effectiveness of ChronoRAG through experiments on the NarrativeQA and GutenQA dataset, showing substantial improvements in tasks requiring both factual identification and comprehension of complex sequential relationships, underscoring that reasoning over temporal order is crucial in resolving narrative QA. [1]

1 Introduction

Long-context question answering tasks, which require the ability to utilize one or more long documents (Pang et al., 2022), present a significant challenge in natural language processing. While modern transformer-based Large Language Models (LLMs) have shown a remarkable ability to handle long contexts (Liu et al., 2025; Wang et al., 2024), they face fundamental limitations when confronted with extremely long-form text. Processing extensive documents for every query leads to major computational inefficiency, and as the context grows longer, the models' ability to accurately identify and prioritize relevant information decreases, impacting the reliability of their outputs.

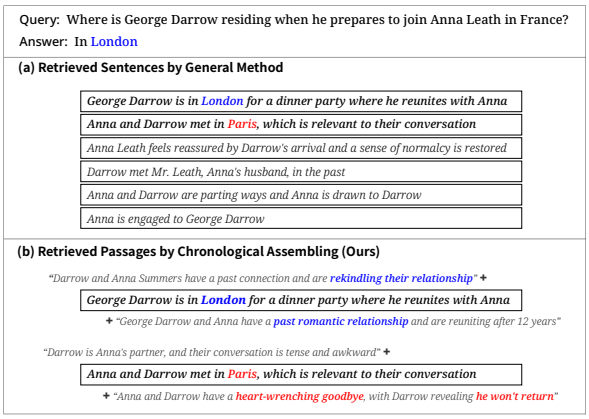

Figure 1: Retrieval comparison for a narrative query. (a) Fine-grained indexing returns six standalone sentences, leaving key clues detached. (b) Our chronological assembling retrieves passages that include their immediate chronological context, preserving the narrative flow. Boxes indicate the directly retrieved sentences.

To address these challenges, Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) has become a standard approach, focusing on efficiently retrieving only relevant segments from large documents to integrate into the model's context window. This selective retrieval method helps models leverage vast knowledge bases far beyond their built-in context limits.

However, a fundamental methodological gap exists in most RAG frameworks (Lewis et al., 2020; Sarthi et al., 2024): they primarily treat documents as a collection of short, independently-retrieved snippets of information. This methodology fundamentally conflicts with the sequential nature of long-form narratives, such as those found in history, literature, and film. Narrative texts are uniquely defined by their structure; they can be **extremely long**, their individual passages often fail to convey

[1] The source code will be released upon paper acceptance.

the full story unless **read in order**, and grasping the **chronological and relational connections between passages is essential** for comprehension. Treating passages as isolated facts severs these critical links, fragmenting the narrative timeline.

Figure 1 illustrates the mismatch between conventional retrieval strategies and the characteristics of narrative data. As shown in (a) of Figure 1, a common approach is to retrieve as many sentences as possible that are likely to match the query based on textual similarity. To do so, documents are typically stored as isolated sentences. While such methods may successfully retrieve a sentence containing the correct answer, they often fail to provide sufficient contextual cues. This can create ambiguity, making it unclear whether "London" or "Paris" is the location relevant to the question, even if both are mentioned in the retrieved results.

To address this issue, we introduce ChronoRAG, a novel RAG-based approach that embodies an alternative strategy grounded in the principle that solving narrative-based problems fundamentally requires recognizing the chronological order of events. Instead of maximizing the number of retrieved sentences, our framework, as shown in (b) of Figure 1, retrieves fewer distinct informational units but includes their **surrounding context** to disambiguate meaning. This approach provides the crucial contextual clues—indicating that "London" is associated with a reunion while "Paris" pertains to a farewell—that are essential for accurate question answering. ChronoRAG achieves this by clarifying dispersed narrative content into structured passages and explicitly capturing the temporal relationships between them, enabling the retrieval of a coherent narrative flow rather than a collection of isolated facts.

We empirically validate our proposed approach on the NarrativeQA (Kočiskỳ et al., 2018) and GutenQA (Duarte et al., 2024). To rigorously test temporal reasoning, we isolate a subset of "Time Questions" that require understanding event sequences. Our experiments show that our method achieves significant improvements in both the complete dataset and the specialized Time Question set. Notably, these results are achieved using lighter graph construction and retrieval mechanisms than those found in existing summary and graph-based methods, demonstrating enhanced performance in identifying individual facts and comprehending complex relational structures.

Our contributions can be summarized as follows:

- We find that resolving narrative QA requires leveraging event chronology and preserving contextual flow, which guides our method in distilling dispersed story elements into coherent, temporally aware passages.
- We introduce a novel RAG framework, ChronoRAG, which refines raw text into structured passages, explicitly maintains temporal links between events, and incorporates adjacent context.
- Experiments demonstrate the effectiveness of our framework, and emphasizing event-to-event relations drives performance gains for both factual and temporal queries, highlighting the critical role of relational understanding over entity extraction.

2 Related Work

Passage Granularity Document indexing approaches have been explored with varying passage granularity to improve retrieval precision. DenseXRetrieval (Chen et al., 2024) advocates finer granularities to enhance information precision. Conversely, MolecularFacts (Gunjal and Durrett, 2024) demonstrates that overly granular decompositions such as atomic facts or propositions often lose critical contextual cues, advocating instead for concise yet contextually coherent units. Our method strikes a balance by using atomic facts as keys for indexing, while preserving broader narrative flows as values during retrieval, thereby combining precision with coherence.

Summary-Based Document Augmentation Summary-based indexing methods construct hierarchical structures by iteratively compressing document segments into progressively shorter summaries. RAPTOR (Sarthi et al., 2024), MemWalker (Chen et al., 2023), and ReadAgent (Lee et al., 2024) demonstrate how such designs enhance retrieval accuracy and contextual coherence by filtering out irrelevant passages and aggregating query-focused summaries. However, the deep hierarchies adopted in these approaches often lead to redundant overlaps and high computational costs. Our approach simplifies the hierarchical concept by adopting a single-layer summary, significantly reducing computation and overlap issues while maintaining contextual effectiveness.

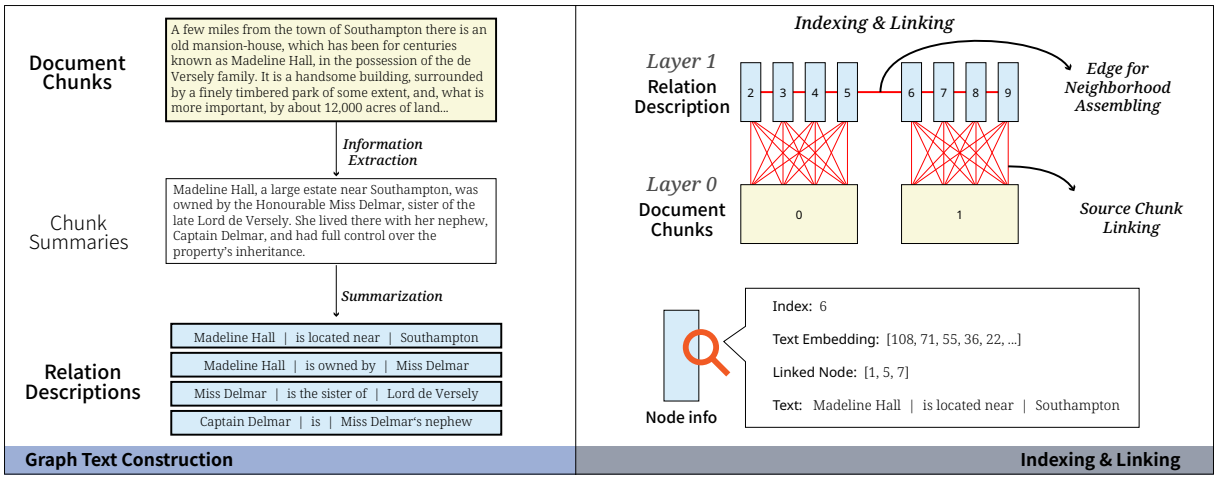

Figure 2: The offline Graph Construction pipeline of ChronoRAG. This process transforms an unstructured narrative document into a structured, two-layer graph that explicitly encodes chronological relationships.

Knowledge Graph-Based Document Augmentation Graph-based augmentation represents another line of work that emphasizes structured relational knowledge. GraphRAG (Edge et al., 2024) formalizes the paradigm through components such as query processors, retrievers, organizers, and generators, and leverages graph traversal and community detection to retrieve information beyond lexical similarity. LightRAG (Guo et al., 2024) reduces preprocessing and latency overhead by encoding relational signals into dense indices and employing coarse-to-fine retrieval. Extensions such as EventRAG (Yang et al., 2025b) construct event knowledge graphs that capture temporal and causal dependencies, while Entity–Event RAG (Zhang et al., 2025) prevents collapsing distinct entity mentions by maintaining separate but linked entity and event subgraphs. Iterative reasoning approaches, such as KG-IRAG (Yang et al., 2025a), further refine this idea by incrementally retrieving over temporal and logical dependencies. Despite these advances, most methods emphasize static entity-centric relations; our framework differs by explicitly modeling sequential narrative relations, thereby addressing a critical gap in capturing dynamic contextual flows.

3 ChronoRAG

We present ChronoRAG, a novel RAG framework specialized for narrative texts where chronological context is crucial. Most RAG systems treat documents as a collection of independent facts, which fragments the timeline and severs the contextual links essential for understanding sequential events. To address this, we design our framework to reconstruct narrative flow by explicitly modeling and preserving the temporal order of events. As described in Figure 2, our framework is composed of two primary stages: an offline **Graph Construction** phase where the original documents are processed into a hierarchical, linked structure, and an online **Passage Retrieval and Answer Generation** phase where the constructed graph is used to answer queries.

3.1 Offline Graph Construction

This offline phase transforms a raw document into a structured, two-layer graph that captures both factual information and narrative chronology. As shown in Figure 2, this graph is composed of two levels: a foundational ***Layer 0*** containing the original document text divided into sequential, fixed-length chunks, which preserves narrative detail, and an abstract ***Layer 1*** built from concise, structured relation descriptions that represent the key events and relationships. The graph construction process involves four steps.

3.1.1 Document Chunking

Due to inherent limitations in processing an entire document simultaneously, we first divide the original document (D) into fixed-length chunks (d_i), each consisting of up to k tokens. This approach ensures that all retrieved document segments fit within a predefined context length, thereby maintaining both the quality and manageability of retrieval results. The document is initially segmented into individual sentences, which are then sequentially appended to each chunk. When the cumulative length exceeds k tokens, the next sentence is assigned to a new chunk, ensuring that sentences are not split across chunks. In rare cases where

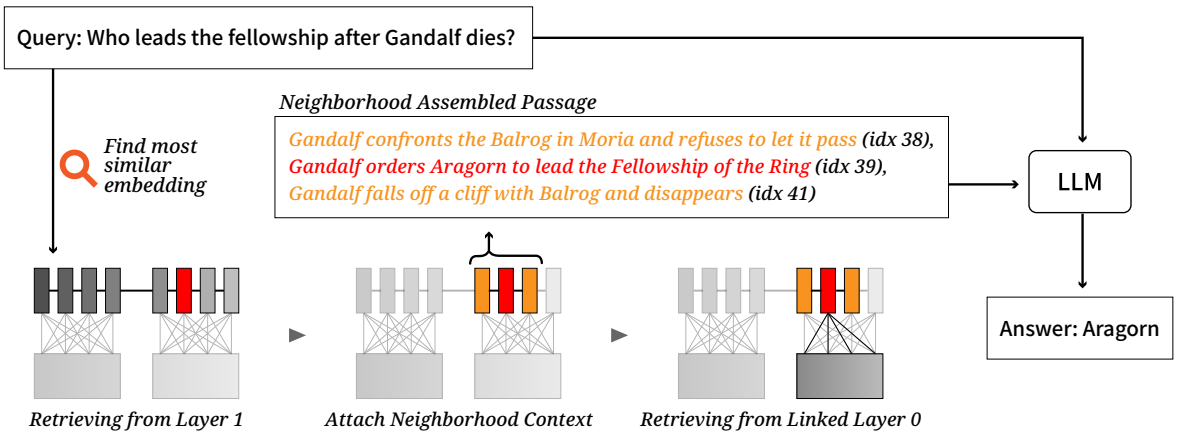

Figure 3: The online Passage Retrieval process of ChronoRAG for a sample query. This demonstrates how the pre-constructed graph is used at inference time to assemble a chronologically coherent context for the LLM.

a single sentence itself exceeds k tokens, the sentence is split to guarantee that every chunk remains within the token limit. These segmented document chunks serve as the fundamental retrieval units and constitute *Layer 0* of the graph constructed in subsequent stages. The document D is thus represented as a set of chunks, where each chunk d_i has a token count less than k, as follows:

$$D = \{d_1, d_2, d_3, \ldots, d_i\}, \quad |d_i| < k$$

3.1.2 Chunk Summarization

Next, we sequentially cluster the chunks in the original document order, grouping every l chunks. We concatenate and summarize each cluster's texts using an LLM. We utilize sequential clustering because it preserves the document's original order while maintaining a controllable and consistent input length for the LLM. This summarization step facilitates higher-level representation learning by focusing on the overall flow of the document rather than retaining excessive local detail. For example, the left panel of Figure 2 shows how a raw *Document Chunk* about *Madeline Hall* is condensed into a more concise *Chunk Summary*. For each cluster of l chunks, we generate a summary S_i by an LLM using a summarization prompt $P_{\text{summarize}}$ as formulated below: (We provide a full prompt in Appendix C.)

$$s_i = \text{LLM}(P_{\text{summarize}}, \{d_{i1}, d_{i2}, d_{i3}, \ldots, d_{il}\})$$

3.1.3 Entity-Relation Extraction

We then transform the generated summaries via LLM into relational descriptions among entities. We adapt the prompt of GraphRAG (Edge et al., 2024) into a one-shot instruction for entity–relation extraction. (Full prompts are in Table 6 of Appendix.) From LLM's outputs, which consist of both entity descriptions(E_i) and relation descriptions(R_i), we only use the relation descriptions to form the *Layer 1* nodes of the graph. This extraction step decomposes the summarized text into retrieval-friendly fragments, as described in the left panel of Figure 2, where the summary is broken down into atomic facts like `"Madeline Hall is owned by Miss Delmar"`.

Here, we exclude entity descriptions because identical entities may appear redundantly across multiple chunks. Finally, we formalize this extraction step, where an LLM processes each summary S_i to produce a set of entities E_i and relations R_i, as shown below: (Full prompts are in Appendix C.)

$$\{E_i, R_i\} = \text{LLM}(P_{\text{extraction}}, S_i)$$

3.1.4 Hierarchical and Temporal Indexing

This step involves assigning indices to the relation description sentences (R_i) derived from the summary and the document chunks (d_i) from the original text. Document chunks are indexed sequentially according to their original order in the source text, while the relation description sentences are indexed either based on the earlier chunks from which they are derived and in the order in which they were generated during information extraction.

Each document chunk corresponds to a *Layer-0* node, and each relation sentence forms a *Layer-1* node. For quick access, each *Layer-1* node connects to its corresponding *Layer-0* nodes—those

within the cluster from which it was derived—by establishing directed edges. Additionally, adjacent *Layer-1* nodes (according to their index) are also linked via edges. This indexing and edge-construction process results in the formation of a unified graph structure.

3.2 Online Passage Retrieval

At inference time, we handle a query through a hierarchical retrieval process that leverages the constructed graph to assemble a rich, chronologically-aware context for the LLM.

3.2.1 Hierarchical Retrieving

We leverage the hierarchical granularity of *Layer 1* and *Layer 0* for retrieval. We begin by retrieving high-precision relation descriptions from *Layer 1* based on semantic similarity to the query. Then, using the links established during indexing, we retrieve the related *Layer 0* chunks to provide a comprehensive and balanced context. As illustrated in Figure 3, this process first identifies a key event in *Layer 1* and later retrieves the detailed source text from *Layer 0*. This step is crucial because *Layer 0* often retains omitted details and original dialogues that are valuable for question answering.

3.2.2 Neighborhood Assembling

We then augment retrieved relational descriptions with their surrounding context to reconstruct a narrative flow. Rather than relying on isolated facts, we aim to provide contextually rich information. As in the example of Figure 3, after ChronoRAG retrieves a key event, such as `"Gandalf orders Aragorn to lead the Fellowship of the Ring (idx 39)"`, the system automatically appends its chronological neighbors, including `"Gandalf confronts the Balrog... (idx 38)"` and `"Gandalf falls off a cliff... (idx 41)"`. This creates a coherent, temporally ordered passage that preserves the local storyline, providing crucial context that isolated facts would lack.

3.2.3 Answer Generation

Finally, we combine the original query with the context obtained through hierarchical retrieval and neighborhood assembling and feed them into the language model. We separate each passage by double line breaks and sort by relevance, enabling accurate and coherent answer generation.

4 Experiments

4.1 Experimental Setup

Dataset We employ the NarrativeQA (Kočiskỳ et al., 2018) and GutenQA (Duarte et al., 2024) datasets to measure ChronoRAG's performance. NarrativeQA comprises 10,557 question-answer pairs from 391 stories, while GutenQA consists of 3,000 pairs from 1,000 stories. To specifically evaluate temporal reasoning, we construct a "Time Questions" subset by selecting all samples containing at least one of a set of predefined temporal keywords: {'When,' 'While,' 'During,' 'After,' 'Before'}. This process yields 1,111 questions from NarrativeQA and 662 from GutenQA, respectively. These questions require retrieving and reasoning over multiple related events, making them a demanding benchmark for temporal understanding.

Evaluation Metric We measure answer quality using ROUGE-L (Lin, 2004), which computes the Longest Common Subsequence (LCS) overlap between a generated answer and its corresponding human reference. Due to the short and pronoun-heavy nature of NarrativeQA answers, ROUGE-L effectively captures agreement in key word sequences without penalizing minor rephrasings.

In addition to ROUGE-L, we employ cosine similarity and LLM-based evaluation to better assess semantic fidelity. ROUGE may fail to capture semantically similar answers that differ lexically, so we incorporate complementary metrics to address this limitation. We compute cosine similarity using the Snowflake model (Merrick et al., 2024), which is also used during the retrieval process. It measures the embedding-based similarity between the generated answer and the reference answer.

For LLM-based evaluation, we use GPT-4.1 Mini (Achiam et al., 2023). Given the question, summary, gold passage, and ground-truth answer, the GPT model performs binary classification—[Correct] or [Wrong]—to determine whether the generated answer can be considered valid. See Appendix C for the detailed prompt.

Baselines We compare against five existing methods that differ in information extraction, representation, and retrieval structure:

- **NaiveRAG:** A standard RAG pipeline that performs chunk-level retrieval only, without further structuring (Lewis et al., 2020).

NarrativeQA						
Metric	ROUGE		CosineSim		LLM Eval (ACC)	
Subset	Whole Data	Time Question	Whole Data	Time Question	Whole Data	Time Question
NaiveRAG	0.255	0.227	0.841	0.844	0.183	0.144
Propositionizer	0.262	0.238	0.846	0.852	0.189	0.141
RAPTOR_CT	0.298	0.262	**0.854**	**0.858**	0.241	0.178
RAPTOR_TT	0.289	0.253	0.851	0.854	0.231	0.170
LightRAG	0.240	0.214	0.841	0.845	0.182	0.123
GraphRAG	0.195	0.185	0.823	0.830	0.139	0.106
ChronoRAG (Ours)	**0.308**	**0.268**	0.853	0.854	**0.257**	**0.195**

GutenQA						
Metric	ROUGE		CosineSim		LLM Eval (ACC)	
Subset	Whole Data	Time Question	Whole Data	Time Question	Whole Data	Time Question
NaiveRAG	**0.166**	**0.172**	0.769	0.778	**0.251**	**0.278**
Propositionizer	0.151	0.142	**0.779**	**0.785**	0.167	0.140
RAPTOR_CT	0.159	0.164	0.775	0.784	0.244	0.269
RAPTOR_TT	0.104	0.102	0.762	0.769	0.134	0.119
LightRAG	0.083	0.083	0.740	0.750	0.075	0.077
GraphRAG	0.122	0.115	0.760	0.767	0.134	0.119
ChronoRAG (Ours)	0.159	0.170	0.776	**0.785**	0.248	0.275

Table 1: QA Performance on NarrativeQA and GutenQA across ROUGE, CosineSim, and LLM Eval metrics. Top performance is bolded, Second best is underlined.

- **RAPTOR:** Clusters semantically similar chunks via embedding similarity and builds a recursive summarization tree over clusters to guide retrieval—CT (Collapsed Tree) flattens each root-to-leaf path into one high-level summary, whereas TT (Tree Traversal) retains the full hierarchy and drills down level-by-level to gather finer-grained context (Sarthi et al., 2024).
- **LightRAG:** Constructs a lightweight entity–relation graph to enable fast context retrieval using dual-level extraction, prioritizing computational efficiency and incremental updates (Guo et al., 2024).
- **GraphRAG:** Builds a richer graph with detailed relation weighting and neighborhood assembly to support deeper multi-hop retrieval, capturing both high-level relation summaries and their underlying chunks (Edge et al., 2024).
- **Propositionizer:** Transforms the entire source text into fine-grained propositions (atomic sentences) and treats each proposition as a retrieval unit, then feeds retrieved propositions into the generation model (Chen et al., 2024).

Implementation Details. All baselines share identical hyperparameter settings: top-k of 20 for retrieval, contextTokenLengthLimit of 1,500 tokens, and the same greedy decoding strategy during generation. We perform all summarization and entity–relation extraction steps with meta-llama-3-8B-Instruct (Grattafiori et al., 2024). We compute retrieval scores using embedding similarity exclusively; we don't use BM25 (Robertson et al., 2009) to prevent distortion of the original text during generation. Specifically, we employ the arctic-Snowflake-embed-l (Merrick et al., 2024) for generating embeddings, and use unifiedqa-v2-t5-3b-1363200 (Khashabi et al., 2022) for final answer generation. All retrieved contexts fed into the generator respect the 1,500-token length limit to ensure fair comparison across all methods.

4.2 Main Results

Performance Comparison Table 1 shows that our proposed ChronoRAG outperforms all baselines on the NarrativeQA dataset, with particularly strong gains on the "Time Question" subset. An analysis of the baselines on this dataset reveals distinct failure modes corresponding to their retrieval strategies. Summarization-based methods like RAPTOR-CT are the next-best performers but still lag our method; RAPTOR's approach of clustering semantically similar chunks is insufficient for narratives, as it can group thematically related but chronologically distant events. Similarly, methods retrieving isolated text units, such as NaiveRAG and Propositionizer, struggle to provide sufficient context and fragment the narrative flow. GraphRAG records the lowest score, as its exhaustive entity–relation extraction adds thousands

of trivial nodes, burying key plot elements under noise and severely diluting precision.

However, on the GutenQA dataset, the results are more nuanced. We explain that this is likely due to the nature of questions in GutenQA, which are often constructed by extracting text directly from the source and thus reward methods with the most direct access to the original passage details. For this reason, methods that heavily refine or abstract the text, such as Propositionizer and the graph-based approaches, perform poorly because they lose the specific passage-level information required. Even RAPTOR is penalized, as its summary-first approach limits direct access to the source text. In contrast, both NaiveRAG and ChronoRAG employ a consistent, fixed-length chunking strategy, which helps normalize the potentially unrefined structure of the GutenQA data. NaiveRAG's slight edge in some metrics can be attributed to its direct retrieval from these unaltered chunks, whereas ChronoRAG's abstraction steps, while beneficial for narrative synthesis, risk filtering out the fine-grained details that these specific questions demand.

Ablation Study We conduct ablation studies to investigate the effectiveness of different components and settings of ChronoRAG. As shown in the Table 2, without summarizing the original text and extracting entity relations shows a significant performance degradation, showing the importance of chunk summarization. The effects of summarization are twofold: it leaves only important information, making retrieval easier, and when assembling, it clarifies the flow. The results without passage assembling are obtained by individually searching for entity relations extracted from the summary, while the results without chunk summarization are obtained by searching for entity relations directly extracted from the 10 chunks. Despite not connecting nearby passages in both settings, a significant performance difference is observed in the Time-Question.

Method	Whole Data	Time Question
ChronoRAG	**0.308**	**0.268**
w/o Passage Assembling	0.295	0.252
w/o Chunk Summarization	0.272	0.233
w/o Relation Extraction	0.255	0.227

Table 2: Ablation study of ChronoRAG's core components on the NarrativeQA dataset, with performance measured by ROUGE-L.

4.3 Analysis

Trade-off between Linking Window and the Number of Retrieved Passage We analyze the trade-off of using a larger linking window for passage assembly. While a wider window provides more local context, it also reduces the number of distinct passages that can be retrieved within a fixed token budget. Our experiment confirms this is detrimental; as shown in Table 3, extending the window to two neighbors ("Extended Link Window") lowers performance on both the whole dataset and the temporal questions subset. This result validates that our default approach of using a more concise, immediately adjacent context is more effective.

NarrativeQA	Whole Data	Time Question
ChronoRAG	**0.308**	**0.268**
Extended Link Window	0.300	0.258
Merged Key	0.302	0.257

Table 3: Analysis of design variations within the ChronoRAG framework on the NarrativeQA dataset measured by ROUGE-L.

Key-Value Separation in Information Retrieval We investigate the effectiveness of key-value retrieval design of ChronoRAG, which separates the precise fact used for retrieval (the key) from the broader context provided to the model (the value). To validate this, we test an alternative "Merged Key" approach where the retrieved fact and its neighbors are combined into a single text unit before retrieval. As shown in Table 3, this modification results in a slight performance decrease, indicating that our key-value separation is an effective strategy for balancing retrieval precision and contextual coherence.

	# of Sentence	mean(Similarity)
Retrieved Sentences	104,981	0.838
Assembled Sentences	209,092	0.785

Table 4: Embedding Similarity Between Query and Retrieved/Assembled Passages.

Effect of Neighborhood Assembling We validate the effectiveness of neighborhood assembling in enriching the retrieved context with information that is chronologically relevant but not necessarily the most semantically similar to the query. To demonstrate this, we compare the average embedding similarity between the query and the initially

Query: Where is George Darrow residing when he prepares to join Anna Leath in France?		
Method	Model Answer	Retrieved Context
(a) ChronoRAG	London	*George Darrow and Anna have a past romantic relationship and are reuniting after 12 years, George Darrow is in London for a dinner party where he reunites with Anna, Darrow and Anna Summers have a past connection and are rekindling their relationship*
(a) RAPTOR	a country estate with his friend Owen.	*The story revolves around George Darrow, a young man who is on leave from his military duties and is staying at a country estate with his friend Owen. Darrow is struggling to come to terms with his feelings for Anna, a woman who...*
(c) LightRAG	athenee	*-----Entities(KG)-----* *Anna is a complex and multifaceted character who is deeply involved in the story. She is the step-mother of Owen....* *-----Relationships(KG)-----* *[{"description": "Anna is concerned about her step-son's departure..."},*
(d) GraphRAG	"DOVER", "GEO"	*-----Entities-----* *0,"RESTAURANT","LOCATION","The restaurant is a location where Anna..."* *1,"DOVER","GEO","Dover is a location where George Darrow takes a train to."* *-----Relationships-----* *0,"CHELSEA","DARROW","Darrow has a past connection with Chelsea...* *1,"ANNA","EFFIE'S EDUCATION","Anna is concerned about Effie's education...*
(e) Propositionizer	Givre	*Anna decided to accompany Darrow to Paris.* *Darrow's disappointment was tempered by the certainty of being with Mrs. Leath again before she left for France.* *Darrow was under the same roof with Anna again.*

Figure 4: A qualitative comparison of retrieved context and model answers for the query, "Where is George Darrow residing when he prepares to join Anna Leath in France?".

retrieved sentences versus the final assembled passages. As shown in Table 4, the average similarity for the assembled passages is discernibly lower than that of the sentences retrieved by similarity alone. This gap indicates that the neighboring passages, while chronologically adjacent, are semantically distinct from the initial query hit. In narrative texts where surrounding content carries strong causal or temporal relevance, this mechanism allows the model to incorporate pertinent information beyond the limits of pure similarity search, thereby improving the context for answer generation.

Case Study Figure 4 presents excerpts of the original passages retrieved by each method for the example shown in Figure 1. RAPTOR retrieves summary passages, which enable access to content covering a wide range of information. However, these summaries frequently include information that is not pertinent to the query, or conversely, omit critical details necessary for answering the question due to length constraints imposed by the summarization process. LightRAG and GraphRAG extract entities and relations directly from the original text. In particular, GraphRAG was found to underperform compared to direct retrieval to the source chunk, likely due to its tendency to include exhaustive explanations of all elements. Propositionizer and LightRAG offer relatively general-level granularity explanations, yet they still struggle to address questions that require understanding the changes in the relationship between Anna and George. In contrast, ChronoRAG identifies the minimal set of chronologically adjacent passages while suppressing unrelated narrative details, illustrating its strength in maintaining temporal coherence and reducing retrieval noise.

Computation Costs Our pipeline is computationally efficient, requiring just two LLM calls per 1,000 tokens for graph construction. Although this cost increases linearly with document length, it remains lower than competing methods like recursive summarization. Furthermore, only one LLM call is required for answer generation during search, with our method still attaining the highest performance despite its efficiency.

5 Conclusion

We present ChronoRAG, an RAG framework that can effectively and efficiently handle narrative text. Our framework refines content through summarization and relation extraction, and improves overall performance through simple passage augmentation that connects adjacent events via an index. This suggests that it is important not only to organize individual events and elements in narrative texts but also to connect events that are spatially and temporally close to each other.

Limitations

Our proposed ChronoRAG framework explicitly models temporal order to improve narrative question answering. However, the approach has several limitations. First, while our graph construction pipeline is lightweight compared to prior graph-based methods, it still requires multiple LLM calls for summarization and relation extraction, which may introduce latency in large-scale deployments. Second, our evaluation focuses primarily on two English narrative datasets (NarrativeQA and GutenQA), and results may not directly generalize to non-English narratives or other domains such as legal or medical texts. Third, although our method improves temporal reasoning, it does not yet capture more complex discourse phenomena such as causal chains spanning distant events. Future work could explore integrating richer discourse structures and extending experiments to multilingual or domain-specific corpora.

Ethics Statement

All experiments in this paper are conducted on publicly available datasets (NarrativeQA and GutenQA) and widely used pre-trained models under their respective licenses. No private or sensitive user data is involved. While narrative corpora are relatively low-risk, they may still reflect historical or cultural biases present in the source texts, which can propagate into retrieval and generation. We report results in aggregate without attempting to infer personal attributes, and we adhere to ethical guidelines for reproducible research by ensuring transparency in data usage and methodology. We also plan to release our implementation to support open and verifiable research practices.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. 2023. Walking down the memory maze: Beyond context limit through interactive reading. *arXiv preprint arXiv:2310.05029*.

Tong Chen, Hongwei Wang, Sihao Chen, Wenhao Yu, Kaixin Ma, Xinran Zhao, Hongming Zhang, and Dong Yu. 2024. Dense x retrieval: What retrieval granularity should we use? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15159–15177.

André V. Duarte, João DS Marques, Miguel Graça, Miguel Freire, Lei Li, and Arlindo L. Oliveira. 2024. LumberChunker: Long-form narrative document segmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6473–6486, Miami, Florida, USA. Association for Computational Linguistics.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Anisha Gunjal and Greg Durrett. 2024. Molecular facts: Desiderata for decontextualization in llm fact verification. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3751–3768.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*.

Daniel Khashabi, Yeganeh Kordi, and Hannaneh Hajishirzi. 2022. Unifiedqa-v2: Stronger generalization via broader cross-format training. *arXiv preprint arXiv:2202.12359*.

Tomáš Kočiskỳ, 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.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. 2024. A human-inspired reading agent with gist memory of very long contexts. In *Proceedings of the 41st International Conference on Machine Learning*, pages 26396–26415.

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

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Jiaheng Liu, Dawei Zhu, Zhiqi Bai, Yancheng He, Huanxuan Liao, Haoran Que, Zekun Wang, Chenchen Zhang, Ge Zhang, Jiebin Zhang, et al. 2025. A comprehensive survey on long context language modeling. *arXiv preprint arXiv:2503.17407*.

Luke Merrick, Danmei Xu, Gaurav Nuti, and Daniel Campos. 2024. Arctic-embed: Scalable, efficient, and accurate text embedding models. *arXiv preprint arXiv:2405.05374*.

Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, et al. 2022. Quality: Question answering with long input texts, yes! In *2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022*, pages 5336–5358. Association for Computational Linguistics (ACL).

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

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

Chonghua Wang, Haodong Duan, Songyang Zhang, Dahua Lin, and Kai Chen. 2024. Ada-leval: Evaluating long-context llms with length-adaptable benchmarks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3712–3724.

Ruiyi Yang, Hao Xue, Imran Razzak, Hakim Hacid, and Flora D Salim. 2025a. Beyond single pass, looping through time: Kg-irag with iterative knowledge retrieval. *arXiv preprint arXiv:2503.14234*.

Zairun Yang, Yilin Wang, Zhengyan Shi, Yuan Yao, Lei Liang, Keyan Ding, Emine Yilmaz, Huajun Chen, and Qiang Zhang. 2025b. Eventrag: Enhancing llm generation with event knowledge graphs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16967–16979.

Ze Yu Zhang, Zitao Li, Yaliang Li, Bolin Ding, and Bryan Kian Hsiang Low. 2025. Respecting temporal-causal consistency: Entity-event knowledge graphs for retrieval-augmented generation. *arXiv preprint arXiv:2506.05939*.

A The use of Large Language Models

We prepared the manuscript independently and used an LLM assistant solely for minor refinement purposes (e.g., clarity improvements and grammar checking). The assistant was not involved in research ideation or content creation. The tool we employed was ChatGPT-5.

B Implementation Details

We conduct our experiments using an AMD EPYC 7313 CPU (3.0 GHz) paired with four NVIDIA RTX 4090 GPUs. We use Python 3.11.5 and PyTorch 2.3.1 for the software environment. We access meta-llama-3-8B-Instruct via the OpenRouter (2025) API with temperature set to 0 (greedy decoding) for generating answers from GutenQA. The detailed hyperparameters used in our experiments can be found in Table 5.

We utilize the nano-graphrag repository[2] and the lightrag repository[3] to implement the GraphRAG and LightRAG baselines, respectively. For proposition generation from the original documents, we employ the chentong00/propositionizer-wiki-flan-t5-large model.

In our experiments, GraphRAG is configured to operate in local mode, while LightRAG is set to hybrid mode. For both models, the retrieval parameter top_k is set to 10.

C Prompts

ChronoRAG As shown in Table 6, we employ task-specific prompts for both summarization and entity–relation extraction. For summarization, the model is instructed to condense each cluster of document chunks into a concise description without pronouns, ensuring that the resulting text remains self-contained. For entity–relation extraction, the model is guided by a structured instruction that requires listing entities with type and description, followed by explicit relationships between entities with a numeric strength score. This structured output is essential for constructing the ChronoRAG graph, as it enables us to represent both the factual content of the story and the temporal or relational dependencies between entities. By combining these two prompts, we distill long narrative texts into coherent graph structures that support accurate and temporally consistent retrieval.

LLM Eval In addition, as shown in Table 7 and Table 8, we assess model outputs with an LLM-based judge. GPT-4.1 Mini is prompted with a structured template that includes the question, a gold passage providing narrative context, the gold answer, and the model-generated answer. The prompt explicitly instructs the judge to output only one of two labels—`[Correct]` or `[Wrong]`—without explanation. This design ensures consistency and avoids subjective variation. The evaluation complements automatic metrics such as ROUGE and embedding-based similarity by correctly recognizing semantically valid answers even when they differ lexically.

D Linking Case Study

As illustrated in Table 9, we present a representative linking case demonstrating how ChronoRAG assembles adjacent passages. Bracketed sentences mark the retrieved evidence aligned with the query, while the surrounding context ensures coherence. This example shows how linking preserves narrative flow and enables the model to answer correctly ("a dog"), reducing ambiguity compared to isolated retrieval.

[2] https://github.com/gusye1234/nano-graphrag
[3] https://github.com/HKUDS/LightRAG

Parameter	Value
max token length in chunk	100
number of cluster in chunk	10
max token length in summarization	2000
max token length in Entity Relation Extraction	2000
do_sample	False
summarization model	meta-llama/Llama-3.1-8B-Instruct
entity relation extraction model	meta-llama/Llama-3.1-8B-Instruct
model for answer generation	unifiedqa-v2-t5-3b- 4281363200
embedding model for text embedding	Snowflake/snowflake-arctic-embed-l
max token length of context	15
retrieving top_k	15
max number of retrieved passage	20

Table 5: Configuration parameters for ChronoRAG pipeline.

Summarization Prompt

System: Write a summary of the following context as short as possible within five sentences. DO NOT USE PRONOUN.

Context: `<document chunk>`

Summary:

Entity–Relation Extraction Prompt

Goal

Given a text document that is potentially relevant to this activity and a list of entity types, identify all entities of those types from the text and all relationships among the identified entities.

Steps

1. Identify all entities. For each identified entity, extract:
   – `entity_name`: Name of the entity, capitalized
   – `entity_type`: One of `[Leading Role, Supporting Role, Object]`
   – `entity_description`: Comprehensive description of the entity's attributes and activities

Format: `("entity"|<entity_name>|<entity_type>|<entity_description>)`

2. From step 1 entities, identify clearly related (source, target) pairs and extract:
   – `source_entity`, `target_entity`
   – `relationship_description`
   – `relationship_strength` (numeric)

Format: `("relationship"|<source_entity>|<target_entity>|<relationship_description>|<relationship_strength>)`

3. Return all items in English as a single list delimited by `&`.
4. Finish with `<End>`.

Example

Text: The Verdantis's Central Institution will meet on Monday and Thursday; a policy decision is due Thursday 1:30 p.m. PDT, followed by a press conference where Chair Martin Smith will take questions. Investors expect the Market Strategy Committee to hold the benchmark rate at 3.5%–3.75%.

Output:

```
("entity"|CENTRAL INSTITUTION|ETC|The Central Institution is the Federal Reserve of Verdantis, setting interest rates on Monday and Thursday)
& ("entity"|MARTIN SMITH|PERSON|Chair of the Central Institution)
& ("entity"|MARKET STRATEGY COMMITTEE|ORGANIZATION|Committee making key decisions about interest rates)
& ("relationship"|MARTIN SMITH|CENTRAL INSTITUTION|Chair will answer questions at a press conference|9)
<End>
```

Real Data

Text: `<summary text>` *Output:*

Table 6: Prompt templates used in ChronoRAG for summarization and entity–relation extraction.

System Prompt
You are the grader who determines the correct answer precisely.
User Prompt
Determine whether the user's answer to the following question is correct or not. Choose only one of the two options: `[Correct]` or `[Wrong]`. Do not explain your reasoning; state only your judgment. - Question: <question> - Summary: "<document_summary>" - Golden answer: <gold_answer> - User's answer: <model_answer> - Judgement:

Table 7: Prompt used for LLM-based evaluation on narrativeQA.

System Prompt
You are the grader who determines the correct answer precisely.
User Prompt
Determine whether the user's answer to the following question is correct or not. Choose only one of the two options: `[Correct]` or `[Wrong]`. Do not explain your reasoning; state only your judgment. - Question: <question> - Literature Context: "<chunk_must_contain>" - Golden answer: <gold_answer> - User's answer: <model_answer> - Judgement:

Table 8: Prompt used for LLM-based evaluation on GutenQA.

Example Linking Case
Correct: [Correct] Question: What kind of animal does Anna have when Gurov first sees her? Golden Answer: [“A dog”, “Small dog”] Generated Answer: a dog Passage: Dmitri Gurov becomes infatuated with the lady with the dog and tries to get to know her, **[Gurov is drawn to Anna’s innocence and vulnerability, and they spend time together]**, Gurov and Anna share a romantic moment, but Anna is overcome with guilt and shame. Kovrin and the Pesotskys are preparing for the wedding, **[Kovrin is deeply in love with Tanya, and their relationship is central to the story]**, The black monk appears to Kovrin, filling him with pride and a sense of exalted consequence, and Kovrin is obsessed with his work. Gurov is drawn to Anna’s innocence and vulnerability, and they spend time together, **[Gurov and Anna share a romantic moment, but Anna is overcome with guilt and shame]**, Anna confesses her infidelity to her husband, feeling she has betrayed him.

Table 9: Representative linking case used in ChronoRAG.