HisGraphRAG: GraphRAG for Vietnamese Historical question answering

Hoang-Thanh Nguyen[1,2], Tung Le[1,2], Huy Tien Nguyen[1,2,*]
[1]Faculty of Information Technology, University of Science, Ho Chi Minh city, Vietnam
[2]Vietnam National University, Ho Chi Minh city, Vietnam
23C11048@student.hcmus.edu.vn, {lttung, ntienhuy}@fit.hcmus.edu.vn
[*]Corresponding author: Huy Tien Nguyen - ntienhuy@fit.hcmus.edu.vn

Abstract

Large Language Models (LLMs) often face issues with factual accuracy and hallucinations, especially in specific domains like history. While Retrieval-Augmented Generation (RAG) and Graph-based RAG (GraphRAG) improve reasoning by integrating external knowledge, applying GraphRAG to historical data presents three key challenges: (1) redundant entities in the knowledge graph; (2) a lack of temporal understanding for historical events; and (3) potential for LLM generative capacity to be diluted by excessive retrieved information. We propose HisGraphRAG, a novel framework designed for historical question answering. HisGraphRAG addresses these issues through entity alignment during graph construction, temporal reasoning and reranking for event contextualization, and filtering of irrelevant information to maintain LLM focus. Evaluated on a Vietnamese history university entrance exam dataset, HisGraphRAG significantly boosts reasoning performance across various LLMs. This work offers a more reliable and effective GraphRAG framework for historical inquiry, enhancing LLM applications in this crucial field.

1 Introduction

Multiple-choice question answering (MCQA) in the history domain poses significant challenges for large language models (LLMs), which often struggle with precise factual recall and chronological reasoning (Brown et al., 2020). This can lead to *hallucination*, the generation of factually incorrect information (Ji et al., 2023), a problem especially prevalent in less-documented historical contexts like Vietnamese history (Kumar and Pavlick, 2022).

Retrieval-Augmented Generation (RAG) mitigates this by supplementing LLMs with external documents, improving factual grounding (Lewis et al., 2020a). However, standard RAG struggles with the complex inter-entity relationships and temporal structures (Zhou et al., 2022) inherent in historical narratives (Lewis et al., 2020b). For instance, answering a question like, "During World War II, what was the impact of 1943 American aid to other countries?" requires reasoning about connections not explicitly present in the retrieved text. To address this, GraphRAG builds structured knowledge graphs into the RAG pipeline, modeling entities and their relationships (Microsoft Research, 2024). While this enhances context, it introduces new limitations in historical domains. First, despite overall improvements, GraphRAG can paradoxically perform worse than a naive LLM in certain cases, as shown in Figure 1. Second, it often creates *duplicated entities* (e.g., "Soviet Union" and "Soviet") during indexing, fragmenting the knowledge graph as detailed in Table 1. Third, it fails to preserve *temporal grounding*, which is crucial for answering time-sensitive questions.

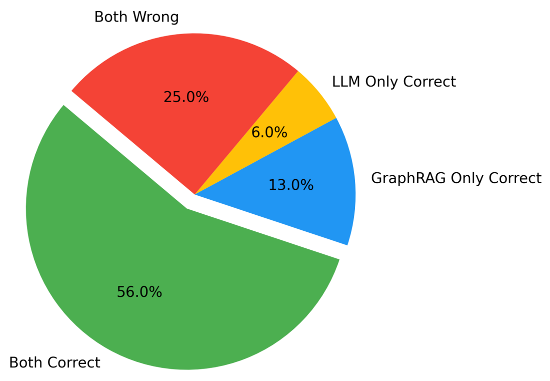

Figure 1: Accuracy Comparison: GraphRAG vs LLM-Only (Vietnamese University Entrance Exam)

To overcome these limitations, we propose HisGraphRAG, a novel framework with three targeted enhancements. First, we implement *entity alignment* during indexing to merge semantically equivalent nodes, creating a cleaner graph. Second, we

	Number of nodes	Number of relations
W/o Entity alignment	1334	681
Entity alignment	1186	510
% reduction	11.09%	25.11%

Table 1: Number of duplicated entity (Building on the summary version of grade 12 Vietnamese history book)

introduce *temporal retrieval*, using time expressions in the query to fetch chronologically relevant events. Finally, inspired by human test-taking strategies, we add an *answer candidate filtering mechanism* to eliminate implausible options before retrieval, sharpening the reasoning focus. Our main contributions are:

- Identify and address three critical limitations of GraphRAG in historical QA: duplicated entities, loss of temporal context, and distraction from irrelevant information.
- Introduce HisGraphRAG, a system that integrates entity alignment, temporal retrieval, and question candidates filtering to improve performance on Vietnamese historical QA. Then we demonstrate that our model's selective reasoning strategies effectively enhance accuracy on university entrance exam datasets.

2 Related work

Traditional RAG methods primarily rely on retrieving relevant text passages based on semantic similarity, but they struggle to capture complex relational and temporal knowledge inherent in domains like history (Procko and Ochoa, 2024). This limitation motivated the development of Graph Retrieval-Augmented Generation (GraphRAG) (Microsoft Research, 2024), which integrates structured knowledge graphs to provide richer, relational context for LLMs. GraphRAG retrieves graph elements such as nodes, triples, or subgraphs relevant to a query, enabling multi-hop reasoning over interconnected entities and relations.

GraphRAG is designed to minimize hallucinations and improve knowledge updates in LLM by integrating information from external graph knowledge (Procko and Ochoa, 2024; Han et al., 2024). G-Retriever (He et al., 2024) uses embedding cosine similarity to identify relevant nodes and edges for the query, then constructs the subgraph using the Prize-Collecting Steiner Tree (PCST) algorithm, which generates an accurate and compact subgraph for generation (He et al., 2024; Yiqian Huang, 2025). The RoG method (Jiang et al., 2024) proposes a "planning–retrieval–reasoning" model, in which reasoning paths are taken according to a pre-determined plan before performing inference. Meanwhile, GNN-RAG (Mavromatis and Karypis, 2024) utilizes Graph Neural Networks to handle the complex structure in the knowledge graph, combining retrieval augmentation techniques to increase the diversity of the input data. However, the performance of these methods depends deeply on the quality of the graph, and when the resulting subgraph contains a lot of noise or is irrelevant, the performance degrades significantly (He et al., 2024).

Despite these advances, GraphRAG faces several challenges. First, retrieving noisy and irrelevant information will downgrade LLM performence. It's researched and improved in Guo and al. (2025) research by adding two stages filtering in querying process. However, this improvement comes at the cost of increased response time in generating the final answer. Our entity alignment method will reduce duplicated or misaligned entities durign indexing step which reduces the knowledge graph size while remaining retrieval effectiveness. Second, temporal information and relationships between historical events are often overlooked, limiting the model's ability to reason about event sequences and causality. Third, excessive reliance on external graph knowledge can suppress the intrinsic reasoning capacity of the base LLM, then degrading overall performance.

3 Method

To address the key challenges identified in our preliminary studies—namely entity duplication, irrelevant retrievals, and lack of temporal reasoning—we propose a new framework **HisGraphRAG** designed specifically for historical question answering with GraphRAG. An overview of our framework is shown in Figure 2, containing three central principles:

- **Entity Alignment:** Redundant or variant entities (e.g., "President Ho Chi Minh" vs. "Ho Chi Minh") dilute knowledge and compromise reasoning. Aligning these into unified representations is essential to preserve semantic coherence and reduce graph fragment.

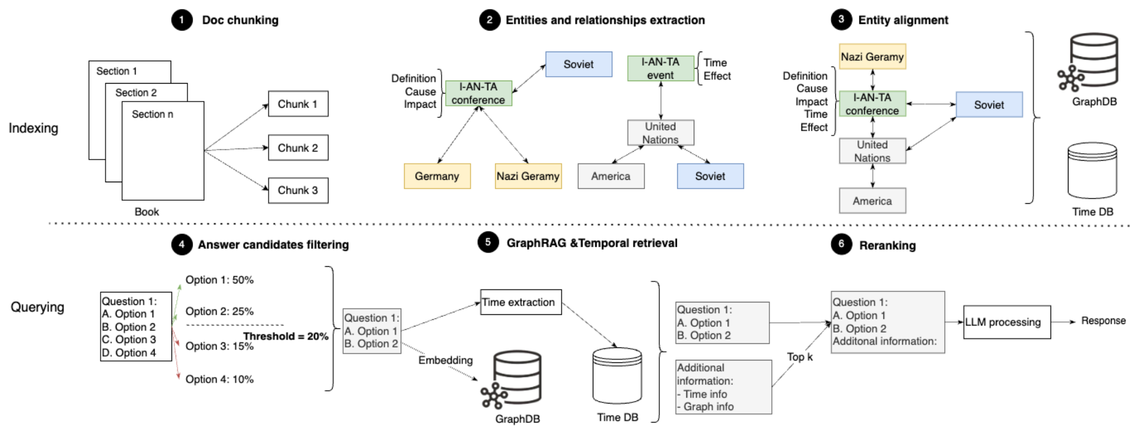

Figure 2: An overview of the HisGraphRAG process, from indexing through querying

- **Answer candidates Filtering:** GraphRAG's retrieval process often includes irrelevant or noisy entries. These not only add confusion but can override otherwise correct LLM responses. Selectively filtering and prioritizing useful information is thus critical.

- **Temporal Awareness:** Historical queries often hinge on timeline-sensitive information. Incorporating temporal cues into retrieval and reranking ensures the selected context aligns with the question's timeframe.

3.1 Doc Chunking

The input dataset for our system comprises history textbooks, which are inherently structured into well-defined chapters and sections. Traditional document chunking strategies—such as segmenting by a fixed number of tokens or characters—fail to preserve the semantic and topical coherence necessary for downstream tasks like retrieval and graph construction. To address this, we adopt a section-aware chunking strategy tailored to the structural granularity of historical texts.

Let a document be composed of a sequence of chapters $C = \{c_1, c_2, ..., c_n\}$, where each chapter c_i contains a sequence of sections $S_i = \{S_{i1}, S_{i2}, ..., S_{im}\}$. Each section S_{ij} is treated as a primary unit of chunking. To maintain context continuity across sections, we apply overlapping context windows by concatenating with a portion of the preceding section when available. Formally, each chunk x_{ij} is generated as:

$$x_{ij} = S_{ij-k} || S_{ij} \tag{1}$$

where k is the overlapping context parameter and || denotes concatenation.

To comply with the LLM input limitations, If the resulting chunk x_{ij} exceeds a token threshold $T_{\max}$, we recursively subdivide it into smaller chunks $x_{ij}^1, x_{ij}^2, \ldots, x_{ij}^l$ such that:

$$\text{len}(x_{ij}^l) \leq T_{\max}, \quad \forall l \tag{2}$$

The history textbooks are provided in PDF format, which includes both text and non-textual elements (e.g., illustrations, charts, and exam questions). We preprocess these documents by converting each page into an image and then employ a visual-language model (VLM), or an LLM equipped with document vision capabilities, to extract textual descriptions from visual elements. This step ensures that all important content—especially figures relevant to historical narratives—is retained in natural language form. At the same time, we filter out irrelevant components, such as quiz questions, using a classification filter tuned to remove interrogative-style sentences. This preprocessing ensures that each document chunk is both semantically cohesive and enriched with textual equivalents of visual data, thus improving the quality of downstream retrieval and reasoning stages.

3.2 Entities and Relationships Extraction

Since our focus is exclusively on historical texts, we design the entity and relationship extraction process to target concepts that are particularly relevant to the domain. Specifically, we extract information related to historical events, key figures, locations, organizations, strategies, and their associated impacts. Each document chunk is processed using a LLM to identify named entities and infer relationships between them. For every pair of related entities, we assign a single relation type: `RELATE_TO`), accompanied by a natural language description that explains the nature of the relationship. This design choice maintains simplicity while still preserving the contextual meaning between entities.

- A graph database that stores entities as nodes and their RELATE_TO relationships as edges, including descriptive annotations.
- Identities was categorized into: person, organization, event, place, action, strategy, impact.
- A SQL database that records temporal attributes of events and entities, enabling chronological filtering in later retrieval stages.

It is important to note that, unlike Microsoft's GraphRAG implementation, we do not include any community reports or crowdsourced annotations. Our method strictly constructs knowledge from the source documents themselves, maintaining a clean and verifiable information graph suitable for history domain.

3.3 Entity Alignment

As history textbooks are organized by chapters and sections, entities representing the same real-world concept (e.g., "Soviet", "Soviet Union") may appear multiple times across different chunks or chapters with slight variations in name or description. If not resolved, this entity duplication introduces noise in the retrieval process—returning redundant or conflicting results—and fragments relationships, weakening the structure of the knowledge graph. To address this, we perform entity alignment using a large language model (LLM) with extended context capabilities. We aggregate entities extracted across all sections and send them in batches to the LLM, which performs a semantic merge of duplicate or near-duplicate entities. For each group of similar entities, the model:

- Selects a canonical name and be validated by LLM then human
- Merges descriptions into a unified summary
- Merges associated relationships while eliminating redundancy

This process ensures that each unique historical figure, event, or organization is represented by a single unified node in the graph, improving both the retrieval precision and the integrity of relational links. By consolidating entity representations and summarizing associated content, the resulting graph becomes more compact, coherent, and interpretable—especially critical for query tasks that involve reasoning over historical timelines or cause-effect chains.

3.4 Answer candidates Filtering

To evaluate the effectiveness of our system, we consider a multiple-choice question-answering setting tailored for historical content. Each question is accompanied by four candidate options: A, B, C, D, with exactly one correct answer. The task is to select the correct option using the knowledge extracted from the document graph. A key challenge arises in the retrieval process: all answer options contribute equally to document retrieval regardless of correctness. This leads to the inclusion of distracting or irrelevant content from incorrect answers, which can degrade both retrieval precision and model confidence. In human examination settings, students naturally focus more on plausible answers while disregarding options that appear clearly incorrect. Inspired by this behavior, we introduce an Answer candidates filtering mechanism that prioritizes likely correct answers.

Given a question q and four answer choices $A = \{a_1, a_2, a_3, a_4\}$, we compute:

$$P(a_i \mid q) = \text{LLM\_score}(q, a_i) \tag{3}$$

Let θ be a confidence threshold. We retain only those options such that:

$$P(a_i \mid q) \geq \theta \tag{4}$$

This filtering strategy offers two major benefits. First, it aligns with the reasoning strategy of proficient human test-takers: before committing to an answer, they mentally discard distractors that are clearly implausible, thereby focusing attention on a narrower and more reliable candidate set. Second, from a technical perspective, the multiple-choice setup inherently introduces redundancy — both between the question and its answer candidates, and among the candidates themselves. Without filtering, this redundancy propagates into the retrieval process, increasing the risk of irrelevant or noisy evidence being incorporated. By retaining only those candidates whose likelihood surpasses the confidence threshold, our method reduces retrieval redundancy and sharpens the focus on semantically meaningful information, ultimately improving both retrieval efficiency and answer accuracy.

3.5 Graph & Temporal Retrieval

In this step, we integrate semantic graph-based retrieval and temporal filtering to enhance the contextual information fed into the final prompt. These two retrieval processes leverage the structured outputs from Step 2: the graph database (containing entities and relationships) and the time-indexed event database.

Graph-based Retrieval:

Given a query q which contains question and answer candidates from Step 4, we embed them into a dense vector $\vec{q}$ using a pre-trained model. Each graph node e_j has an embedding $\vec{e}_j$. Compute similarity:

$$v_q = Embedded(q) \tag{5}$$

Simultaneously, each entity node e in the graph database is stored with its own semantic embedding v_e, computed from both the entity's name and its description. Retrieval is performed by computing similarity scores:

$$\text{sim}(q, e_j) = \cos(\vec{v_q}, \vec{v_e}) \tag{6}$$

Entities with similarity above a retrieval threshold δ are selected:

$$\varepsilon_{retrieved} = \{e \in \mathcal{E} \mid \text{sim}(q, e) \geq \delta\} \tag{7}$$

For each retrieved entity e, we also extract its associated relationships and descriptions to construct a subgraph relevant to the question context

Temporal Retrieval:

To complement graph-based retrieval, we perform time-aware filtering based on temporal cues in the query. If the query q contains a time reference (e.g., a year or period), we extract it and define a time window:

$$[t_{start}, t_{end}] = [t_q - \Delta t, t_q + \Delta t] \tag{8}$$

Finally, the union of retrieved graph-based and time-based information:

$$C_q = \varepsilon_{graph} \cup \varepsilon_{temporal} \cup \varepsilon_{textchunk} \tag{9}$$

3.6 Reranking

The retrieval stage often results in a large set of semantically and temporally relevant entities, relationships, and events. However, not all of these contribute equally to answering the given question. Including too many can overwhelm the prompt and reduce the effectiveness of the language model's reasoning. To ensure focus and precision, we introduce a reranking mechanism driven by the LLM itself.

Instead of relying on a traditional scoring function or trained reranker, we leverage the LLM's internal reasoning to identify the most relevant context items. Specifically, we prompt the LLM to act as a selector, choosing a subset of the retrieved content that is most likely to support answering the question.

Given a query q and retrieved context $C_q = \{c_1, c_2, ..., c_n\}$, we prompt the LLM to select the most relevant k items:

$$C_q^{\text{top-k}} = \text{Select}(q, C_q, k) \tag{10}$$

where k is typically set to a small number to maintain prompt compactness.

This LLM-guided reranking allows for context-aware prioritization, capturing subtle relevance signals that might be missed by static similarity metrics. The selected top-k entities and facts are then composed into the final input prompt for the answering model, significantly improving focus and factual consistency in the output.

4 Experiment

4.1 Dataset

To evaluate the effectiveness of our GraphRAG-based retrieval and reasoning framework in the domain of historical question answering, we construct both a retrieval corpus and a evaluation dataset.

Retrieval Corpus: We use the official Grade 12 Vietnamese History textbook, which serves as the knowledge base for information retrieval and graph construction. The textbook is representative of high school-level historical content and provides rich, structured information in the form of chapters and sections, including text, figures, and timelines.

Evaluation Dataset: For the downstream question answering task, we use real-world standardized multiple-choice questions from the Vietnamese University Entrance Exams in the years 2017 and 2018. For each year, we include three official exam codes, resulting in a total of 120 multiple-choice questions (3 sets × 40 questions) called VUEE-2017 and VUEE-2018. Each question consists of one correct answer and three distractors, following the standard national testing format.

4.2 Experimental setting

We implement our GraphRAG-based system using a modular architecture combining large language models, graph databases, and vector search engines. The key components and hyperparameters are configured as follows:

- Main Language Model: We use ChatGPT-4o mini as the primary model for all reasoning tasks, including entity alignment, reranking, and final answer generation. Its ability to handle long-context inputs and follow prompt instructions makes it well-suited for this structured retrieval setting.
- Embedding Model: For dense retrieval and semantic similarity computation, we use the text-embedding-3-small model. All entities and question inputs are embedded using this model and stored for efficient nearest-neighbor search.
- Graph Database: We store structured entities and their relationships using Neo4j, which enables traversal and context construction via `RELATE_TO` edges.
- Vector Database: All embeddings are stored and queried through Milvus, which supports high-performance similarity search over dense vector representations.
  - Temporal Window: For time-based filtering, we extract time expressions from the query and apply a ±1 year buffer to retrieve relevant historical events.
  - Answer Filtering Threshold: For question option filtering (Step 4), we discard any answer option with a model-estimated confidence below 15%.
  - Reranking Top-k: After retrieval, we prompt the LLM to select the top 15 most relevant context items for final reasoning.
- Implementation Base: Our system builds upon and extends the open-source GraphRAG implementation from the public repository [1].

[1] https://github.com/ImmortalDemonGod/nanographrag

5 Result

5.1 Indexing result

To evaluate the efficiency cost of the proposed entity alignment step, we measured token usage and processing time during the indexing phase, both with and without alignment.

Cost	w/o entity alignment	with entity alignment
In token consumption	605k	669k in
Out token consumption	190k	219k out
Time consumption	322s	270s

Table 2: LLM consumption during indexing step

As shown in Table 2, applying entity alignment leads to a modest increase in input token consumption from 605k to 669k tokens, an approximate 10.6% increase. Similarly, output token usage increases from 190k to 219k tokens, reflecting the additional summarization and relational consolidation introduced during alignment. Despite this increase in token count, the overall time consumption decreases slightly, from 322 to 270 seconds, possibly due to the more compact and structured representation of merged entities reducing downstream processing complexity.

5.2 Querying results

We evaluate the effectiveness of our proposed system, HisGraphRAG, along with several baseline methods and incremental variations, on two standardized historical question-answering benchmarks: VUEE-2017 and VUEE-2018. Table 3 reports the accuracy, token consumption, and response time across all methods.

For the baseline comparison, the LLM-only setting—where the model answers without any external retrieval—achieves the lowest accuracy on both datasets (68.3% for VUEE-2017 and 50.8% for VUEE-2018), indicating the insufficiency of parametric knowledge alone for detailed historical reasoning. When retrieval is added in a naive form via NaiveRAG, which lacks any structured understanding of entities or relationships, performance improves modestly (72.5% and 54.2%), suggesting that even simple information augmentation can be beneficial. These baselines provide a reference point for the gains enabled by structured and filtered retrieval.

When moving to GraphRAG, which incorporates entity-level structure and relationship reasoning, the accuracy improves significantly to 77.5% for VUEE-2017 and 60.0% for VUEE-2018. Adding

Method	VUEE-2017			VUEE-2018		
	Acc	Token	Time	Acc	Token	Time
LLM-only	68.3%	22k	80s	50.8%	23k	80s
NaiveRAG	72.5%	102k	145s	54.2%	95k	150s
GraphRAG	77.5%	920k	210s	60.0%	907k	218s
GraphRAG + Entity alignment	76.7%	1.4M	377s	61.7%	1.4M	385s
GraphRAG + Temporal & Rerank	73.3%	1.7M	927s	56.7%	1.7M	773s
GraphRAG + Answer candidates Filter	80.8%	949k	384s	63.3%	983k	360s
HisGraphRAG (Ours)	**81.7%**	**2.2M**	**977s**	**70.8%**	**1.6M**	**922s**

Table 3: HisGraphRAG accuracy result on VUEE-2017 and VUEE-2018 dataset

entity alignment to GraphRAG yields mixed results: a slight decrease in 2017 (76.7%) but a notable improvement in 2018 (61.7%), indicating that alignment becomes more impactful when the dataset contains more ambiguity or entity duplication. Interestingly, adding temporal retrieval and reranking in isolation causes a decline in accuracy (73.3% and 56.7%), likely because this configuration pulls in excess temporal content not always directly relevant to the question—overloading the prompt and reducing model focus. However, when question filtering is applied alone, accuracy increases sharply (80.8% and 63.3%), highlighting the importance of removing low-confidence distractors before retrieval. Ultimately, the HisGraphRAG system, which integrates all components—including alignment, temporal retrieval, reranking, and filtering—achieves the highest accuracy on both datasets (81.7% and 70.8%).

While **HisGraphRAG** achieves the best results, it does so with increased computational cost. The number of tokens consumed rises to 2.2M and 1.6M respectively for VUEE-2017 and VUEE-2018, and average response time approaches 900 seconds. These costs reflect the additional reasoning steps introduced in the pipeline, but are justified in scenarios where accuracy and context richness are more important than latency. Notably, all systems perform worse on VUEE-2018 than on VUEE-2017, which is consistent with historical data showing that the 2018 exam included more difficult questions, as reflected in average student scores. Even under these conditions, HisGraphRAG maintains strong performance margins, reinforcing its robustness across varying levels of difficulty.

5.3 Error analysis

We analyze one illustrative example from our test data to demonstrate how each component of our method contributes to improved accuracy.

Example Question:

At the *Second National Congress* (February 1951), the *Indochinese Communist Party* decided to establish in each country of Indochina:

A. **Marxist-Leninist Party (Correct)**

B. Coalition Government

C. United Front

D. Armed Force

We summarize the impact of each component in Table 4.

Our proposed method, **HisGraphRAG**, significantly improves the precision and relevance of retrieval by incorporating three key strategies:

- **Entity Alignment** resolves redundancies by merging semantically identical entities into canonical representations, thus eliminating unnecessary duplication and optimizing prompt usage.
- **Temporal Retrieval** leverages temporal context to filter information, constraining retrieved historical events within a precise chronological window relevant to the question.
- **Answer Candidate Filtering** systematically estimates the probability of correctness for each potential answer choice, proactively removing irrelevant distractors to avoid misleading the model.

Component	Without our Method (GraphRAG)	With our Method (HisGraphRAG)
Entity Alignment	Retrieved redundant entities *"Indochinese Communist Party"* and *"Communist Party of Indochina"*, wasting prompt space and causing duplication.	Merged duplicate entities into a single canonical form *"Indochinese Communist Party"*, reducing redundancy and increase useful relationships.
Temporal Retrieval	Retrieved irrelevant congresses (*First* and *Third*), confusing the model with unrelated decisions.	Extracted date (1951) and retrieved events within ±1 year only, filtering out irrelevant historical congresses.
Answer Candidate Filtering	Included distracting information about *Coalition Governments* formed in different historical contexts, misleading the model toward incorrect Option B.	Filtered out irrelevant candidates by estimating answer probabilities, ensuring only contextually relevant evidence is included.

Table 4: Comparison of outputs with and without key components of our proposed method (HisGraphRAG).

Together, these improvements collectively narrow evidence to highly relevant, concise, and contextually precise content, markedly enhancing the probability of correctly identifying the output—**Marxist-Leninist Party** instead of United Front (answer from GraphRAG). acknowledgement

6 Conclusion

In this work, we presented HisGraphRAG, a novel retrieval-augmented generation (RAG) framework specifically designed to address the challenges of historical multiple-choice question answering. Our method incorporates four key components—entity alignment, temporal-aware retrieval, contextual reranking, and candidate answer filtering—to deliver more accurate and contextually grounded responses. By aligning temporal and entity-level information through structured graph-based knowledge and targeted retrieval strategies, HisGraphRAG significantly enhances the reasoning capability of large language models in historical domains. Compared to standard RAG and GraphRAG baselines, our approach achieves notable gains in accuracy, demonstrating its ability to reduce confusion caused by duplicated entities, temporal inconsistencies, and irrelevant distractors. These findings underscore the critical importance of integrating structured semantic knowledge and temporal constraints to support complex historical understanding and decision-making.

While HisGraphRAG shows promising improvements, it also increases computational cost due to longer query times and higher token consumption. In future work, we plan to optimize the temporal retrieval process, explore more efficient indexing strategies, and develop scalable entity alignment techniques, such as hybrid embedding methods and rule-based clustering. These enhancements will help make the approach faster and more practical for large-scale and real-time applications.

7 Acknowledgement

This research is partially funded by the Vingroup Innovation Foundation (VINIF) under the grant number VINIF.2021.JM01.N2

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, and 1 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Kai Guo and al. 2025. Empowering GraphRAG with Knowledge Filtering and Integration. *arXiv preprint arXiv:2503.13804v1*.

Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, and et al. 2024. Retrieval-augmented generation with graphs (graphrag). *arXiv preprint arXiv:2501.00309*.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. Poster #4607, East Exhibit Hall A-C.

Zhijing Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yujin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys (CSUR)*, 55(12):1–38.

Boran Jiang, Yuqi Wang, Yi Luo, Dawei He, Peng Cheng, and Liangcai Gao. 2024. Reasoning on efficient knowledge paths: Knowledge graph guides large language model for domain question answering. *2024 IEEE International Conference on Knowledge Graph (ICKG)*, pages 142–149.

Shailza Kumar and Ellie Pavlick. 2022. Revisiting zero-shot question answering performance of multilingual language models on low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3669–3679.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Douwe Kiela, Anurag Batra, Tim Rocktäschel, and Sebastian Riedel. 2020a. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Mandar Kulkarni, Mike Luo, Ian Wigham, Daniel Beck, and 1 others. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Costas Mavromatis and George Karypis. 2024. Gnn-rag: Graph neural retrieval for large language model reasoning. *ICLR 2025 Conference*.

Microsoft Research. 2024. GraphRAG: Graph-Augmented Retrieval for LLMs. `https://microsoft.github.io/graphrag/`.

Tyler Thomas Procko and Omar Ochoa. 2024. Graph retrieval-augmented generation for large language models: A survey. *2024 Conference on AI, Science, Engineering, and Technology (AIxSET)*, pages 166–169.

Xiaokui Xiao Yiqian Huang, Shiqi Zhang. 2025. A cost-efficient multi-granular indexing framework for graph-rag. *arXiv preprint arXiv:2502.09304*.

Yichong Zhou, Mo Yu, Tongtao Zhao, Huan Sun, and Claire Cardie. 2022. Temporal question answering with sequential temporal graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 740–753.