2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)

Law GraphRAG: An Advanced Legal Question-Answering System

Haoxing Zhai
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
nightskyzhx@zju.edu.cn

***Abstract*—The emergence of large language models (LLMs) has transformed natural language understanding and generation, enabling diverse applications such as conversational agents and automated summarization. However, LLMs often struggle with hallucinated or factually incorrect outputs, particularly when addressing global summary questions requiring comprehensive document-level context. Retrieval-augmented generation (RAG) enhances LLMs by incorporating external knowledge, but its local retrieval focus limits its effectiveness for tasks like Query-Focused Summarization (QFS). To address this, we propose Law GraphRAG, a graph-based extension of RAG that enables comprehensive analysis of legal document collections. By constructing a graph-based text index, Law GraphRAG captures intricate interconnections within legal texts, such as precedents, amendments, and cross-references, providing a richer contextual foundation for answering complex legal questions. Our approach leverages graph structures for both indexing and retrieval, scaling the analysis to the full document corpus while maintaining precision and contextual relevance. Using a dataset of Chinese data compliance laws and regulations, experimental results demonstrate significant improvements on comprehensiveness and effectivess of generated answers. Law GraphRAG offers a powerful tool for advancing intelligent legal information retrieval and summarization.**

Keywords—large language models, retrieval-augmented generation, graph-based retrieval, query-focused summarization, knowledge graphs

I. INTRODUCTION

The development of large language models (LLMs), such as the GPT [1], [2], Llama [3] and Deepseek [4] series, has revolutionized natural language understanding and generation, enabling applications such as conversational agents, automated summarization, and domain-specific question answering. Despite their remarkable capabilities, LLMs often produce hallucinated or factually incorrect outputs, particularly when faced with incomplete or ambiguous context.

To overcome this challenge, retrieval-augmented generation(RAG) [5], which combines LLMs with external retrievalsystems, was proposed. By retrieving relevant documents ortext chunks and incorporating them into the model's input,RAG enables LLMs to generate answers grounded in externalknowledge sources. However, RAG fails to answer globalsummary questions over document collections, particularlyon query-focused summarization (QFS) tasks [6], since it isdesigned to answer questions that are contained in local textregions. To overcome this limitation, GraphRAG [7], [8], agraph-based text indexing approach, has been proposed. Byleveraging a graph-based text index, Graph RAG extends theapplicability of RAG to global summary questions acrossdocument collections, demonstrating improved performance in terms of the comprehensiveness and diversity of generated answers.

This approach is particularly impactful in the legal field. Legal documents, such as statutes, regulations, and case laws, often have intricate interconnections and dependencies that are difficult to capture using linear retrieval methods. Traditional retrieval techniques frequently struggle with maintaining the depth of these relationships, as they rely primarily on keyword matching rather than understanding the nuanced legal context. By employing a graph structure, GraphRAG can effectively model these relationships, such as precedents, amendments, and cross-references, providing a richer contextual understanding. Unlike conventional retrieval models, which may overlook indirect but legally significant connections, the graph-based approach ensures that all relevant legal principles are incorporated into the response. This enables the generation of responses or summaries that are not only accurate but also contextually grounded, ensuring relevance to complex legal queries. Moreover, this structured approach minimizes the risk of misinterpretation by linking related legal texts in a coherent and logically consistent manner. Law GraphRAG facilitates answering complex legal questions by scaling the scope of analysis to the entire document corpus while maintaining the precision required for legal interpretation.

Law GraphRAG employs a two-stage process: graph construction and retrieval from the graph as shown in Fig. 1. On indexing time, Law GraphRAG constructs a graph from laws and regulations. This process involves parsing large volumes of legal text, extracting entities, identifying relationships, and structuring the data into an interconnected legal knowledge graph. Fig. 2 shows an example of the constructed graph. On query time, it generates answers utilizing information retrieved from the graph. By leveraging this structured retrieval system, the model can dynamically adjust to different query types, ensuring flexibility in handling both narrow and broad legal inquiries. The dataset used contains laws and regulations of data compliance in China. Our results demonstrate evident improvements in the comprehensiveness and effectiveness of generated answers, positioning Law GraphRAG as a valuable tool for advancing intelligent legal information retrieval and summarization.

979-8-3315-0976-7/25/$31.00 ©2025 IEEE

II. Related Work

Retrieval-augmented generation [5] is a method to anwer user questions over entire datasets, without the need for additional model training. RAG methods enhance the performance of LLMs on user questions over specific datasets. According to this survey [9], the development of RAG can be categorized into 3 stages: Naive RAG, Advanced RAG, and Modular RAG. However, RAG fails to answer global summary questions over document collections since it is designed to answer questions that are contained in local text regions.

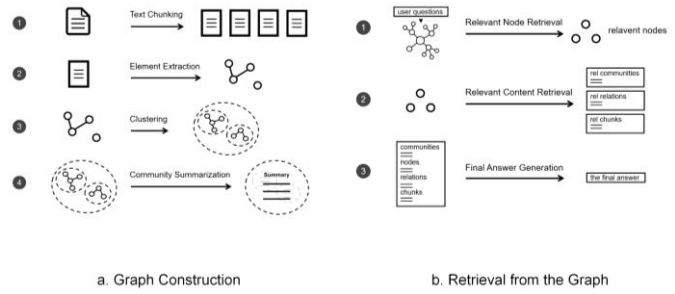

Fig. 1. Law GraphRAG framework.

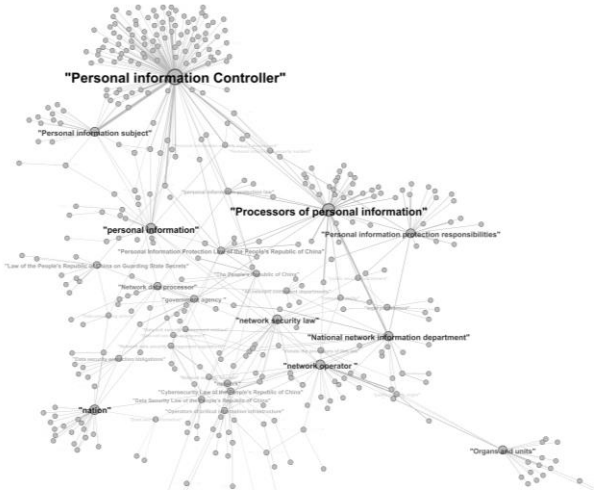

Fig. 2. An example of the constructed graph.

Some graph-based methods have been proposed to improve the reasoning ability of large language models over structured data in a unified way. Think-on-Graph framework [10] integrates large language models and knowledge graphs for improved reasoning tasks. StructGPT [11] implements an Iterative Reading-then-Reasoning (IRR) framework that uses specialized interfaces to access and manipulate structured data. GNN-RAG [12] is a novel framework combining Graph Neural Networks (GNNs) and Large Language Models, employing GNNs for reasoning over dense KG subgraphs to retrieve answer candidates and reasoning paths.

Compared to text-based RAG, Graph RAG [7], [8] takes into account the relationships between texts and incorporates the structural information as additional knowledge beyond text, enhancing query-focused summarization over large datasets. There has already been some research[13], [14] on the application of Graph RAG in Specific Domains.

III. Methodology

A. Graph Construction

Step 1: Text Chunking Laws and regulations often contain long text content, and some of them might greatly exceed the context limitations of LLMs. Some legal documents can greatly exceed these context boundaries, making it difficult for LLMs to capture all the relevant information at once. Expanding the window can still lead to issues such as the model losing critical details, especially in the middle portions of lengthy contexts[15], where important nuances or connections might be overlooked. The raw text is split into chunks by token size to maintain appropriate context windows.

Step 2: Element Extraction A multi-step prompt is used to extract entity instances and relation instances from each chunk. Fig. 3 shows an example of the prompt and results of entity extraction. These elements are then utilized to construct an undirected weighted graph. First identify all entities and extract their name, type and description. Then identify all pairs of entities that are clearly related to each other and extract the source entity, target entity, relationship description and relationship strength for each of the relations. Before inserting elements into the graph, we need to merge entity nodes with the same name and relation edges with the same source and destination nodes. This is performed through LLM summarization over the descriptions of all the matched element instances. The nodes are also embedded into an entity vector database for query time use.

Step 3: Clustering The goal is to partition the graph into communities of nodes with stronger connections to one another than to the other nodes in the graph. Here we use Leiden[16] algorithm to construct hierarchical community structure efficiently. Each level of this hierarchy provides a community partition that covers the nodes of the graph in a mutually-exclusive, collective-exhaustive way. This hierar- chical approach provides insights into the graph structure at varying levels of granularity, enabling divide-and-conquer global summary.

Step 4: Community Summarization For each community, we create a report-like summary to get an overview of all the entities and relations in this community. The report consists of title, summary, impact severity rating, rating explanation and detailed findings. The context provided for report generation is as follows:

- **Node Data:** Node data includes id, entity, type, description, and degree. Sort the nodes by degree and truncate the node data list by token size.
- **Edge Data:** Edge data includes id, source, target, description, and degree. Sort the edges by rank (degree) and truncate the edge data by token size.
- **Sub-Community Data:** If node data or edge data is truncated and the community involves sub-communities, or it’s set to force the use of sub-community data, use sub-community data instead. For each sub-community, pack its id, report, rating and occurrence to form a sub-community data list. Truncate the list by token size. If there are still tokens left under context limit, append node

data and edge data, prioritizing the nodes and edges not included in the truncated list of sub-communities.

B. Retrieval From The Graph

Step 1: Relevant Node Retrieval When querying, the user's question is passed along with the conversation history to Graph RAG. Here we use a bottom-up retrieval strategy to enhance accuracy and directness. Query the entity vector database to retrieve the top-k relevant nodes to the question with cosine similarity as relevance metric, and gather their node data from the graph.

Step 2: Relevant Content Retrieval We need to retrieve relevant communities, text chunks and edges to the nodes.

- **Communities:** Retrieve all the communities in certain level range that the relevant nodes belong to, sort them by occurrence and rating, and truncate the community data list by token size.

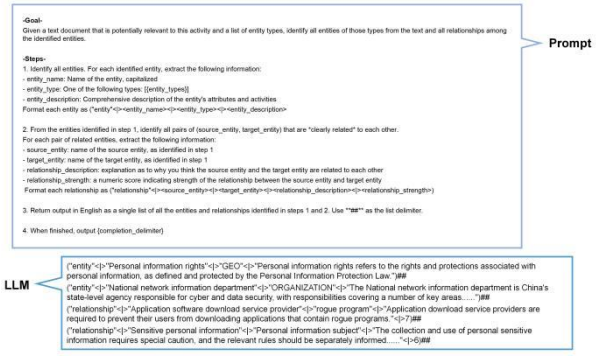

Fig. 3. An example of the prompt and results of entity extraction.

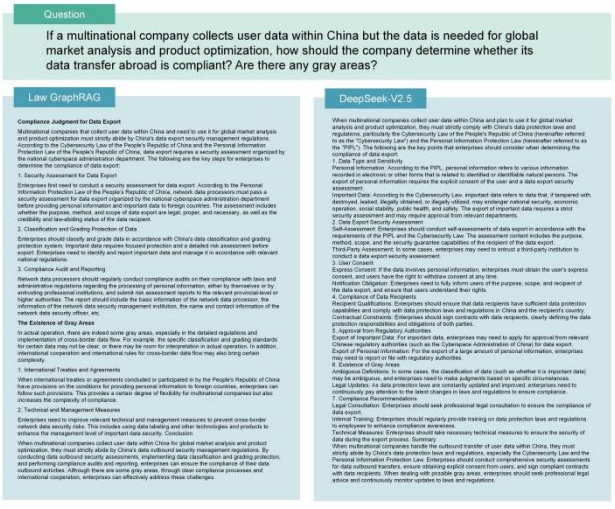

Fig. 4. An example of the outputs of Law Graph RAG and DeepSeek-V2.5.

- **Text Chunks:** Retrieve all the source text chunks of the relevant nodes. Sort them by the order of their related nodes and their relation count to the "one-hop" nodes of the relevant nodes, and truncate the text chunk data list by token size.
- **Edges:** Retrieve all the edges adjacent to the relevant nodes, sort them by rank (degree) and weight, and truncate the edge data list by token size.

Step 3: Final Answer Generation The context including data of relevant nodes, edges, text chunks and communities from the first two steps is constructed and used to generate the final answer.

IV. Experiments

A. Dataset

The dataset used here contains laws and regulations of data compliance of China. These laws and regulations are currently in force and are collected from the Internet. The laws include Personal Information Protection Law, Data Security Law, Cybersecurity Law, and Law on Guarding State Secrets. The regulations include Information Security Technology - Specification for Personal Information Security, Opinions of the Central Committee of the Communist Party of China and the State Council on Building a Basic System for Data and Better Leveraging the Role of Data as an Element of Production, Measures for Security Assessment of Cross-border Data Transfers, and Regulations on the Security Management of Network Data.

TABLE I. Win rates (%) of Law Graph RAG over naive LLMs

Model	Clarity	Comprehensiveness	Effectiveness
DeepSeek-V2.5	2	78	83
GPT-4o mini	1	86	80
Claude 3 Haiku	1	94	81

B. Models

DeepSeek-V2.5: DeepSeek-v2.5 is a LLM model based on the combination of DeepSeek V2 Chat and DeepSeek Coder V2 models. The new model significantly surpasses the previous versions in both general capabilities and code abilities. Chinese comprehensive ability (AlignBench) of the model ranks at the top of the open-source list. The model has a context window of 128K tokens. This is one of the alternative models for graph construction and retrieval of Law Graph RAG.

GPT-4o mini: GPT-4o mini is an affordable and intelligent model for fast, lightweight tasks developed by OpenAI. GPT-4o mini scores 82% on MMLU and currently outperforms GPT-4 on chat preferences in LMSYS leaderboard. The model has a context window of 128K tokens. This is one of the alternative models for graph construction and retrieval of Law Graph RAG.

Claude 3 Haiku: Claude 3 Haiku is a LLM model developed by Anthropic. With state-of-the-art vision capabilities and strong performance on industry benchmarks, Claude 3 Haiku is a versatile solution for a wide range of enterprise applications. Claude 3 Haiku is three times faster than its peers for the vast majority of workloads, processing 21K tokens (~30 pages) per second for prompts under 32K tokens. It also generates swift output, enabling responsive, engaging chat experiences and the execution of many small tasks in tandem. This is one of the alternative models for graph construction and retrieval of Law Graph RAG.

Text-embedding-3-small: text-embedding-3-small is a highly efficient embedding model developed by OpenAI, which provides a significant upgrade over its predecessor, the text-embedding-ada-002 model. This model is used to embed entities in the entity vector database.

C. Results

We choose 3 key dimensions as the criteria for the quality of outputs: clarity, comprehensiveness, and effective-ness. Clarity measures how well the generated responses are articulated, focusing on multiple linguistic factors, including grammatical correctness, coherence, and ease of under-standing. A response exhibiting high clarity should be free from syntactic errors, well-structured, and logically organized, ensuring that it is accessible even to non-expert readers. Comprehensiveness measures the degree to which the output addresses the query in full, ensuring that all relevant aspects of the request are covered without significant omissions or superfluous information. A comprehensive response must demonstrate a thorough understanding of the query by incorporating all necessary details while maintaining conciseness and relevance. Effectiveness measures the practical utility of the response, determining whether the generated content meets the user's intent and provides actionable or insightful information. An effective response not only delivers accurate content but also enhances decision-making by offering well-substantiated reasoning and practical recommendations.

We utilize LLMs to generate a set of 100 diverse and representative legal questions based on the dataset. These questions are carefully designed to encompass a broad range of legal topics, ensuring coverage of different case scenarios and legal principles. They are subsequently used to query both Law GraphRAG and naive LLMs under identical conditions to maintain experimental consistency. In each group of experiments, Law GraphRAG employs the same LLM for both graph construction and retrieval, allowing for a controlled comparison of its performance against conventional LLM-based approaches. The resulting answers are evaluated using LLMs to determine which model performs better in the chosen dimensions.

Fig. 4 shows an example of the outputs of Law Graph RAG and DeepSeek-V2.5. Table I shows the results of the experiments. The experiment results demonstrate that Law Graph RAG performs better in comprehensiveness and effectiveness of generated answers. Its ability to integrate and reason over structured data from the graph enables more complete and contextually relevant responses. However, the study also reveals that Graph RAG falls short in terms of clarity. While the integration of information retrieved from the graph ensures factual accuracy, it can lead to less polished expression at some cases, reducing the clarity of answers. This tradeoff suggests that while Graph RAG excels at providing thorough and useful content, further refinement is needed to enhance the readability and accessibility of its outputs.

V. Conclusion

We introduce Law GraphRAG, which facilitates answering complex legal questions by scaling the scope of analysis to the entire document corpus while maintaining the precision required for legal interpretation. This approach enables legal professionals, researchers, and policymakers to access more comprehensive and contextually relevant infor-mation, allowing for a more informed decision-making. On indexing time, Law GraphRAG constructs a graph from laws and regulations. On query time, it generates context-aware answers utilizing information retrieved from the graph. Our experiments show substantial improvements over naive LLMs on both comprehensiveness and effectiveness of the generated answers, positioning Law GraphRAG as a valuable tool for advancing intelligent legal information retrieval and summarization.

References

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., "Gpt-4 technical report," arXiv preprint arXiv:2303.08774, 2023.

[2] T. B. Brown, "Language models are few-shot learners," arXiv preprint arXiv:2005.14165, 2020.

[3] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., "Llama 2: Open foundation and fine-tuned chat models," arXiv preprint arXiv:2307.09288, 2023.

[4] X. Bi, D. Chen, G. Chen, S. Chen, D. Dai, C. Deng, H. Ding, K. Dong, Q. Du, Z. Fu, et al., "Deepseek llm: Scaling open-source language models with longtermism," arXiv preprint arXiv:2401.02954, 2024.

[5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. K ¨ uttler, M. Lewis, W.-t. Yih, T. Rockt ¨ aschel, et al., "Retrieval-augmented generation for knowledge-intensive nlp tasks," Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020.

[6] H. T. Dang, "Duc 2005: Evaluation of question-focused summarization systems," in Proceedings of the Workshop on Task-Focused Summarization and Question Answering, pp. 48–55, 2006.

[7] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson, "From local to global: A graph rag approach to query-focused summarization," arXiv preprint arXiv:2404.16130, 2024.

[8] Y. Hu, Z. Lei, Z. Zhang, B. Pan, C. Ling, and L. Zhao, "Grag: Graph retrieval-augmented generation," arXiv preprint arXiv:2405.16506, 2024.

[9] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, and H. Wang, "Retrieval-augmented generation for large language models: A survey," arXiv preprint arXiv:2312.10997, 2023.

[10] J. Sun, C. Xu, L. Tang, S. Wang, C. Lin, Y. Gong, H.-Y. Shum, and J. Guo, "Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph," arXiv preprint arXiv:2307.07697, 2023.

[11] J. Jiang, K. Zhou, Z. Dong, K. Ye, W. X. Zhao, and J.-R. Wen, "Structgpt: A general framework for large language model to reason over structured data," arXiv preprint arXiv:2305.09645, 2023.

[12] C. Mavromatis and G. Karypis, "Gnn-rag: Graph neural retrieval for large language model reasoning," arXiv preprint arXiv:2405.20139, 2024.

[13] J. Wu, J. Zhu, Y. Qi, J. Chen, M. Xu, F. Menolascina, and V. Grau, "Medical graph rag: Towards safe medical large language model via graph retrieval-augmented generation," arXiv preprint arXiv:2408.04187, 2024.

[14] Z. Xu, M. J. Cruz, M. Guevara, T. Wang, M. Deshpande, X. Wang, and Z. Li, "Retrieval-augmented generation with knowledge graphs for customer service question answering," in Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2905–2909, 2024.

[15] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," Transactions of the Association for Computational Linguistics, vol. 12, pp. 157–173, 2024.

[16] V. A. Traag, L. Waltman, and N. J. Van Eck, "From louvain to leiden: guaranteeing well-connected communities," Scientific reports, vol. 9, no. 1, pp. 1–12, 2019.