Mid-Term report

Inter-IIT Tech. meet 13.0
Pathway

Dynamic Agentic Retrieval-Augmented Generation with Pathway

Submitted By:
Team - 91

Contents

1 Approach Outline

1.1 Problem Understanding

In the evolving field of Generative AI, Retrieval-Augmented Generation (RAG) systems must address complex queries efficiently and autonomously. The Agentic RAG system aims to extend RAG capabilities by adding autonomy and adaptability in managing information retrieval, data synthesis, and decision-making. The primary challenges in deploying such an agent include:

1. **Autonomous Decision-Making**
   For effective responses, the agent must dynamically determine the best retrieval and response generation strategies. This involves analyzing query complexity, identifying relevant sources, and autonomously choosing the appropriate retrieval method. The agent must also optimize token usage to ensure cost-effective processing without sacrificing accuracy.

2. **Contextual Relevance and Accuracy**
   The agent needs to retrieve and synthesize information from multiple data sources while maintaining high contextual relevance. It must filter, rank, and refine retrieved information to ensure responses are accurate and aligned with the query context, especially when handling varied and unstructured data.

3. **Error Resilience and Adaptability**
   A resilient RAG system must handle potential errors in external API calls or retrieval processes. If a primary retrieval source fails, the agent should automatically switch to alternative sources or strategies to maintain performance and reliability. This adaptability is critical to ensure continuous system functionality under diverse operating conditions.

4. **Resource Efficiency and Scalability**
   The agent must be designed for efficient use of computational resources, particularly when handling large data volumes or high query frequencies. Effective load balancing, caching mechanisms, and optimized retrieval pathways are essential to ensure the system can scale and operate cost-effectively in real-time environments.

This project aims to develop an Agentic RAG system using Pathway that addresses the challenges above. The solution is designed with scalability and efficiency in mind, implementing core functionalities that include:

1. **Dynamic Retrieval Strategy Selection** : The agent assesses query needs and selects the most relevant retrieval approach dynamically, balancing accuracy and cost.

2. **Corrective RAG and Multi-Agent Collaboration** : The system leverages corrective RAG techniques and multi-agent collaboration to enhance response quality and reliability, especially when complex queries span multiple topics.

3. **Error Handling and Service Redundancy** : To ensure resilience, the agent is equipped to detect and respond to errors by switching between available retrieval options and fallback mechanisms.

4. **Real-Time Data Integration and Response Generation** : The agent continuously adapts to incoming data streams, processing information in real-time to provide updated, contextually accurate responses.

1.2 Use-Case Selection and Novelty

1.2.1 Interpretation of Legislative Changes with Contextual Impact Analysis

Our use case aims to introduces a sophisticated RAG system that not only updates legal databases with new legislation but also performs a deep contextual analysis of how these changes interrelate with existing laws, historical judgments, and ongoing cases. **The novelty lies in its ability to map interconnected legal principles and predict the ripple effects of legislative amendments across various legal**

domains. Implementing this requires the development of a comprehensive legal ontology, advanced contextual algorithms, and cross-jurisdictional data integration, enabling proactive strategy adjustments for legal professionals.

1.2.2 Case Preparation

Enhancing case preparation, our solution will leverage RAG to synthesize diverse information sources, including legal precedents, client-specific contexts, and emerging legal theories. The system dynamically adapts to the unique aspects of each case, providing multi-faceted insights and trend analysis to predict potential outcomes. **The innovative aspect is its ability to integrate structured and unstructured data in real-time, offering tailored, data-driven strategic support.** This requires sophisticated data integration techniques, dynamic contextual adaptation, and customizable frameworks to meet specific client needs.

1.2.3 AI-Driven Mediation and Conflict Resolution Assistance Summary

This use case revolutionizes mediation by incorporating an advanced RAG system that offers data-driven insights to facilitate objective and effective dispute resolutions. The system analyzes historical mediation data to identify successful patterns and provides real-time suggestions during sessions, enhancing efficiency and fairness. **The novelty lies in its real-time analytical capabilities and the integration of nuanced human factors into the mediation process.** Implementing this requires extensive pattern recognition algorithms, real-time data processing, and the ability to generate actionable recommendations based on complex qualitative data.

1.2.4 Automated Contractual Risk Assessment and Optimization Summary

Focusing on contract management, our solution will employ RAG to conduct in-depth risk assessments by contextualizing contract terms within the current legal and market environment. The system analyzes contract language alongside external data such as litigation trends and regulatory changes to identify and optimize non-standard clauses. **The innovative feature is its ability to provide context-aware risk evaluations and generate optimized contract language tailored to prevailing conditions.** Implementing this requires advanced natural language understanding, real-time data integration, and adaptive learning mechanisms to continuously refine risk assessments and optimization suggestions.

1.2.5 Innovative Chunking Methods for Structured Tables

Traditional chunking methods often fail to preserve the integrity of structured content, such as tables, by treating them the same as regular text. This can lead to significant misinterpretations, particularly in financial documents where tables are crucial for conveying precise data and relationships. To address this, our RAG solution innovatively treats tables as separate, distinct chunks rather than merging them with other text. This approach significantly improves precision and recall by maintaining the original format and structure of the data. Additionally, while alternative methods like element-based chunking using vision models can better preserve document organization, they introduce considerable processing time. By prioritizing efficiency, our solution effectively retains key structural information without the overhead, ensuring accurate and reliable handling of high-stakes, structured information in financial documents.

1.3 Solution Overview

The proposed solution utilizes two vector databases to store the corpus of laws, the penal code, and past court case rulings. Additionally, AI agents are employed to retrieve specific legal sections, similar cases, and shared components to address the following:

- **Applicable laws for a given case**
- **Cases similar to the given case**

- **Similar components in the cases**

This approach provides a detailed case review that includes key insights, legal interpretations, and observed discrepancies, aiding informed decision-making for the current case.

1.3.1 Assumptions

- The analysis is based on the Constitution and the Penal Code as the primary sources of applicable laws.
- A corpus of Court rulings is referenced to identify precedent cases.

1.3.2 Core Concepts

- **Relevant Laws**: Identification of applicable statutes, regulations, and legal principles that form the foundational framework for legal interpretation in the current case.
- **Similar Cases**: Examination of precedent cases with comparable circumstances to gain insights from judicial decisions, interpretations, and persuasive authority, enhancing case understanding.
- **Similar Components in Cases**: Analysis of common factors—such as involved parties, legal arguments, evidence types, and outcomes—to discern patterns and inform potential case decision-making.

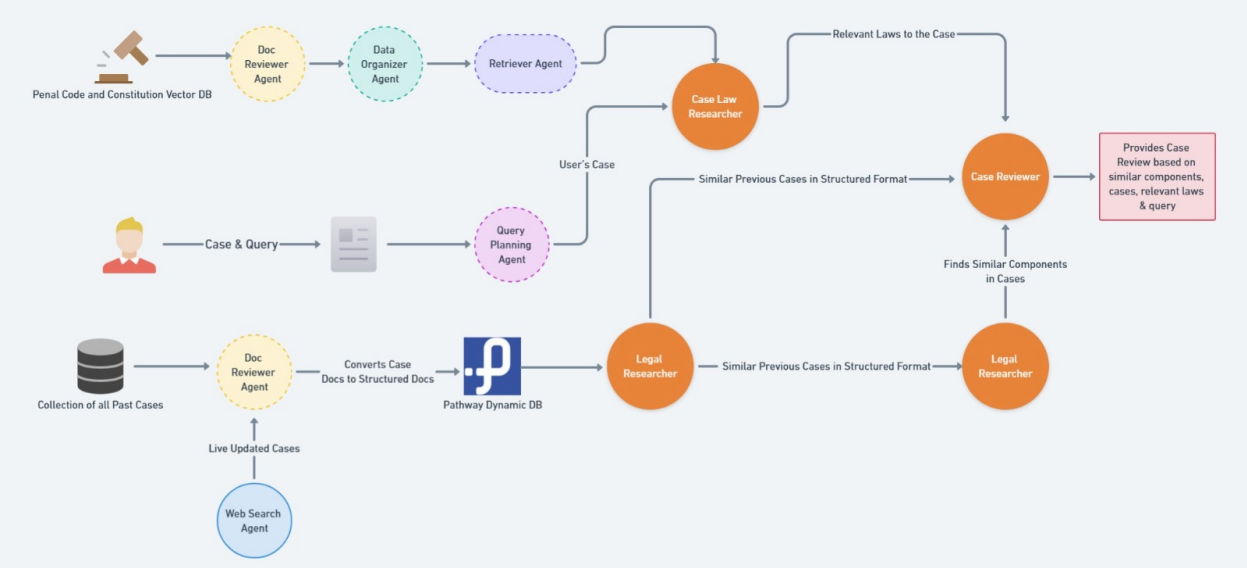
1.3.3 Explanation of Approach

- Two vector databases are employed: one for the corpus of relevant laws and penal codes, and the other for the repository of past cases.
- AI agents are deployed to retrieve pertinent sections of law, analogous legal provisions, and similar case components.
- A comprehensive case review is generated, providing a detailed analysis of key insights and any discrepancies found in the retrieved data.

2 Analysis and Initial Experiments

2.1 System Architecture

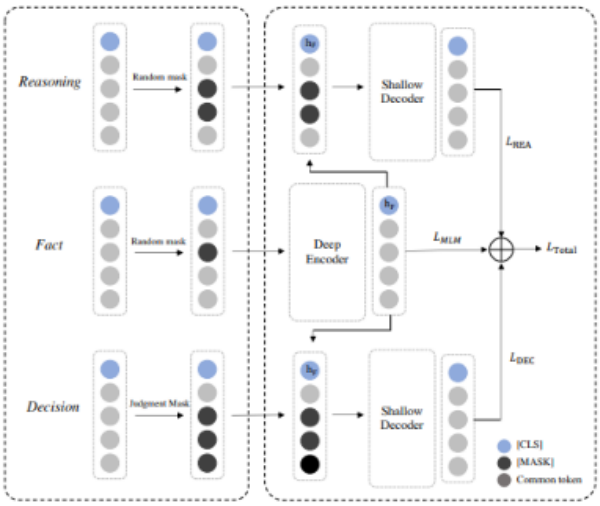
This dynamic agentic RAG system is designed to assist lawyers in various novel legal use cases and legal analysis by orchestrating a series of specialized agents that interact with each other to extract, organize, and evaluate relevant legal information. The workflow begins with the **Doc Reviewer Agent**, which accesses both the *Penal Code and Constitution Vector DB* and a *Collection of all Past Cases*. This agent processes legal documents, transforming them into structured formats. A **Data Organizer Agent** then refines and categorizes information from the Penal Code and Constitution, while the **Query Planning Agent** refines the user's case and query for targeted information retrieval. A **Web Search Agent** supplements this process by fetching live updates for recent cases, contributing to an evolving knowledge base in the *Pathway Dynamic DB*.



With structured documents and user-specific information, the **Case Law Researcher** and **Legal Researcher** agents identify and retrieve similar past cases and relevant laws. The Case Law Researcher specifically handles legal precedents, providing relevant laws related to the user's case. Meanwhile, the Legal Researcher identifies cases with comparable components and structures, ensuring a comprehensive analysis. The results converge at the **Case Reviewer**, which synthesizes insights by evaluating similar case components, laws, and the user's specific query. This agent produces a final review, helping users make informed legal decisions by providing contextual case analysis, relevant laws, and structured legal precedents.

2.1.1 Agents Description

- **Legal Researcher Agent**: This agent introduces citation handling as a novel feature, enabling it to process legal documents with deeper contextual awareness by integrating citations from relevant cases. It leverages an encoder-decoder architecture that reflects the structured nature of legal cases (fact presentation, inferential reasoning, and final verdicts) and uses a *deep-encoder-shallow-decoder* design to enhance retrieval based on distinct case elements.



- **Case Comparator Agent**: Receives relevant cases from the Legal Researcher Agent and performs retrieval based on:

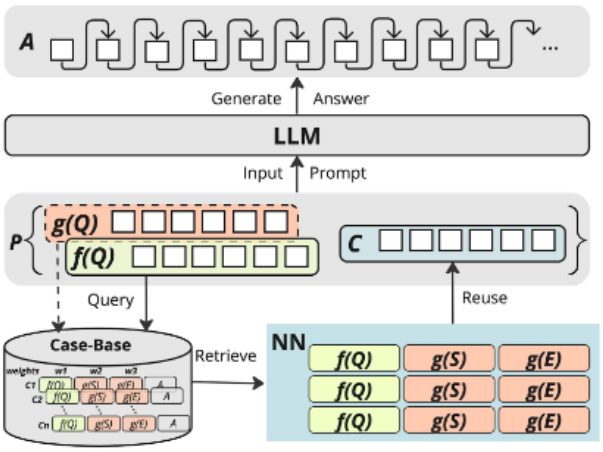
– **Intra Embeddings** $f(.)$: Optimized for attribute matching, enabling local comparisons between similar attributes.

$$\beta_k = \underset{c_i \in C,\, k}{\text{top-k}} \; \text{Sim}(f(Q), f(Q_i))$$

– **Inter Embeddings** $g(.)$: Designed for IR scenarios, allowing inter-attribute similarity assessment for flexible comparisons.

$$\beta_k = \underset{c_i \in C,\, k}{\text{top-k}} \; \text{Sim}(g(Q), g(X_i))$$

Retrieved chunks and documents are passed to the LLM for comparing components and determining outcomes.



- **Document Reviewer Agent**: Utilizes the LexNLP library to convert unstructured legal case documents into structured formats.
- **Web Search Agent**: Performs web searches to extract the latest legal documents and updates from court cases.
- **Data Organizer Agent**: Utilizes *Pathway Vector DB* to dynamically store structured legal case data, updating with new documents as they are received.
- **Case Reviewer Agent**: Synthesizes case review information based on similar components, cases, laws, and the user's query.
- **Retriever Agent**: Uses *Disk ANN* to retrieve chunks from the Data Organizer Agent's Pathway Vector DB.
- **Case Law Researcher Agent**: Returns laws and articles from the constitution and penal code relevant to the user's case.

2.2 Implementation Progress

- **DiskANN**: Discrete Approximate Nearest Neighbor, is a highly optimized indexing algorithm designed to enable efficient and rapid retrieval of nearest neighbors within large datasets. This algorithm applies advanced quantization techniques to represent data as discrete nodes. These nodes are subsequently clustered into shards through the use of the K-means algorithm, optimizing spatial representation and minimizing redundancy.Each query is initially directed to a base shard, which indexes relevant nearby centers. This process allows the query to access multiple overlapping clusters, thus supporting ultra-fast similarity search in latent space.

- **QnA RAG using Gemini and Pathway Vectorstore**: Set up a real-time question-answering pipeline using Pathway and the Gemini model. It streams unstructured data from a specified dropbox folder, performs data chunking, and stores vectorized data within Pathway's vector database. Embeddings are generated through the Gemini embedding model, and the gemini-1.5-flash language model processes user queries by retrieving and analyzing relevant document segments, providing contextually accurate answers based on the indexed content.

- **Querying JSON documents using Mistral**: Set up a RAG pipeline with Pathway vectorstore to query and retrieve information from JSON documents. Documents are loaded from JSON, embedded using GIST-small-Embedding-v0, and indexed with a vector storage system. The mistral model deployed via Ollama processes user queries by retrieving relevant document segments, generating JSON-formatted answer.

- **LangChain chunking methods** : Experimented with various PDF data parsers available in the LangChain documentation. The image below summarizes our findings, detailing the time taken by each method and the granularity of the resulting chunks. This comparison highlights each parser's performance and precision, providing insight into the most efficient methods for handling structured PDF data.

```
CharacterTextSplitter: 0.0003 seconds, 1 chunks
RecursiveCharacterTextSplitter: 0.0081 seconds, 393 chunks
TokenTextSplitter: 0.0417 seconds, 86 chunks
SpacyTextSplitter: 7.0315 seconds, 398 chunks
NLTKTextSplitter: 0.0906 seconds, 394 chunks
MarkdownTextSplitter: 0.0090 seconds, 393 chunks
LatexTextSplitter: 0.0593 seconds, 433 chunks
PythonCodeTextSplitter: 0.0090 seconds, 393 chunks
KonlpyTextSplitter: 0.6144 seconds, 1 chunks
SentenceTransformersTokenTextSplitter: 0.8686 seconds, 194 chunks
```

Figure 1: Chunking Comparisons

2.3 Issues Faced

The following issues were encountered while handling large, complex, interrelated, and ever-evolving legal data within the constitution, penal code, and court rulings:

- The legal framework is continuously updated with new laws and court rulings, which renders pretrained LLMs and traditional RAG frameworks less effective due to their limited ability to update frequently.

- Legal documents are often unstructured and vary significantly in size, making it challenging to apply a common template or structure for organizing them.

- Legal documents are interconnected across both contextual and temporal domains, necessitating systems capable of linking and leveraging information from various documents while maintaining awareness of their temporal relationships.

2.4 Preliminary Results

- **DiscANN with Pathway's vectorstore**: Implemented for dynamic QnA processing using Mistral-based RAG. Pathway's vectorstore efficiently handled dynamic embeddings, adapting to diverse query types and complex data inputs.
- **Chunking and Segmentation Improvements**: Initial tests revealed limitations in retrieval speed and accuracy due to chunking. We refined our methodology, optimizing segmentation strategies to balance precision and efficiency, which improved system response time and result relevance.
- **Optimization for Large Datasets**: To minimize costs while maintaining performance, we integrated caching strategies and optimized query handling in discANN, significantly reducing response latency and computational load.

3 Technical Challenges and Next Steps

3.1 Challenges

- **Hallucinations:** The sources stress that AI systems, even those using retrieval augmented generation (RAG), frequently generate incorrect or misleading information, often referred to as hallucinations. Hallucinations in AI legal research tools primarily manifest in two forms:
  - **Incorrect statements:** AI systems may generate factually inaccurate information, contradicting established legal precedents or presenting fabricated legal concepts.
  - **Misgrounded citations:** AI systems may cite sources that are irrelevant to the query, misinterpret the cited material, or reference sources that have been overturned or are otherwise inapplicable.
- **Retrieval Limitations:** Locating relevant legal information for a given query poses a significant challenge for AI systems. The sources explain that legal research often requires understanding complex legal concepts, interpreting nuanced language, and accounting for the dynamic evolution of case law. Traditional search techniques, such as keyword matching or vector-based similarity, often fall short in accurately identifying the most relevant legal documents.
- **Reasoning Deficiencies:** The sources emphasize that legal reasoning involves multi-step processes, abstract thinking, and contextual understanding that are not easily replicated by current AI systems. While RAG can assist in providing relevant information, AI systems struggle to synthesize information, draw inferences, and apply legal principles to specific situations. For example, evaluating whether a cited case has been overturned or superseded requires sophisticated reasoning abilities that AI currently lacks.

3.2 Future Development Plan

3.2.1 Abstract Representation and Reasoning Over Legal Concepts

A key future objective involves creating abstract representations of fundamental legal concepts, such as murder, rape, theft, etc., to enable higher-level reasoning within the legal domain. This approach would allow the system to generalize specific legal instances to overarching principles and refine its ability to reason about these concepts contextually. Abstracting legal terms will enhance the RAG system's interpretative capabilities and improve its application across cases that involve nuanced or related scenarios, especially in instances where reasoning is necessary to differentiate or align cases based on precedent or legal principles.

3.2.2 Development of a Hierarchical Knowledge Representation

To further enrich the legal reasoning capabilities, a more complex and hierarchical knowledge representation method will be designed. This will involve organizing laws in a structured, multi-tiered framework that

reflects their interdependencies, hierarchies, and cross-references. By capturing relationships—such as dependencies, conflicts, and hierarchical structures—between statutes, regulations, and precedents, the system will be better positioned to explore intricate legal connections, identify relevant precedents, and handle compound queries. This enriched knowledge structure will allow the system to navigate and interpret complex legal queries more effectively.

3.2.3 Indexing and Referencing the Entire Constitution

In order to enhance the depth and reliability of legal referencing, a comprehensive indexing of the entire constitution will be undertaken. By embedding each constitutional clause and making it searchable, the system will gain the capability to reference the foundational legal document dynamically. This will provide users with contextualized insights into how specific legal statutes relate to constitutional principles, strengthening the system's interpretive power in scenarios that require constitutional backing. Comprehensive constitutional referencing will enhance the legal system's relevance in high-level legal analyses and ensure exhaustive legal compliance in the output generated.

4 Conclusion

This project outlines a dynamic, agentic RAG system designed for the legal domain, with capabilities for contextual accuracy, error resilience, and scalability. Key use cases include legislative change impact analysis, AI-assisted case preparation, contract risk assessment, and mediation support. The system leverages innovative chunking for structured data handling, efficient retrieval strategies, and corrective mechanisms to address legal research challenges like hallucinations, retrieval limitations, and reasoning deficiencies. Future developments will focus on abstracting legal concepts, developing a hierarchical knowledge representation, and indexing constitutional clauses for comprehensive legal referencing, enhancing the system's interpretive and reasoning capabilities across complex legal queries.

5 Background Study

References

[1] *CBR-RAG*: https://arxiv.org/html/2404.04302v1

[2] *Legal Bench RAG*: https://arxiv.org/html/2408.10343v1

[3] *Tree RAG*: https://ipchimp.co.uk/2024/02/16/rag-for-legal-documents/

[4] *DiskANN: Fast Accurate Billion-point Nearest Neighbor Search on a Single Node*: https://suhasjs.github.io/files/diskann_neurips19.pdf

[5] *FreshDiskANN for Similarity Search*: https://arxiv.org/pdf/2105.09613

[6] *SAILER: Structure-aware Pre-trained Language Model for Legal Case Retrieval*: https://arxiv.org/abs/2304.11370

[7] *Legal Case Document Similarity*: https://arxiv.org/pdf/2209.12474

[8] *Financial Report Chunking for Effective RAG*: https://arxiv.org/pdf/2402.05131

[9] *LawGPT 1.0: A Virtual Legal Assistant Based on GPT-3*: https://arxiv.org/pdf/2302.05729

[10] *Lawyer LLaMA: Enhancing LLMs with Legal Knowledge*: https://arxiv.org/pdf/2305.15062

[11] *Corrective RAG* : https://arxiv.org/pdf/2401.15884

[12] *Adaptive RAG* : https://arxiv.org/abs/2403.14403

[13] *Light RAG*: https://arxiv.org/abs/2410.05779