VIETNAM GENERAL CONFEDERATION OF LABOUR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY

Lieu Dang Khoa - 520K0140
Nguyen Minh Chi - 52000189

BUILDING A RAG BASED QUESTION–ANSWERING MODEL FOR VIETNAM CIVIL CODE

UNDERGRADUATE THESIS OF COMPUTER SCIENCE

HO CHI MINH CITY, YEAR 2024

VIETNAM GENERAL CONFEDERATION OF LABOUR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY

Lieu Dang Khoa - 520K0140
Nguyen Minh Chi - 52000189

BUILDING A RAG BASED QUESTION–ANSWERING MODEL FOR VIETNAM CIVIL CODE

UNDERGRADUATE THESIS OF COMPUTER SCIENCE

Advised by
Dr. Tran Luong Quoc Dai

HO CHI MINH CITY, YEAR 2024

ACKNOWLEDGEMENT

We would like to express our gratitude to our teacher Dr. Tran Luong Quoc Dai for his guidance throughout the project and his willingness to provide us with valuable research materials. We are also grateful to our parents for their support and our friends for their advice to complete this study. This project has provided us with valuable skills, and we anticipate applying them effectively in the future.

Ho Chi Minh city, 22th August 2024.

Author

(Signature and full name)

Khoa

Lieu Dang Khoa

Nguyen Minh Chi

This thesis was carried out at Ton Duc Thang University.

Advisor: D r . Tran Luong Quoc Dai

(Title, full name and signature)

....

This thesis is defended at the Undergraduate Thesis Examination Committee was hold at Ton Duc Thang University on … /…/……

Confirmation of the Chairman of the Undergraduate Thesis Examination Committee and the Dean of the faculty after receiving the modified thesis (if any).

CHAIRMAN	**DEAN OF FACULTY**
…………………………….	…………………………………

DECLARATION OF AUTHORSHIP

We hereby declare that this thesis was carried out by ourselves under the guidance and supervision of Dr. Tran Luong Quoc Dai and that the work and the results contained in it are original and have not been submitted anywhere for any previous purposes. The data and figures presented in this thesis are for analysis, comments, and evaluations from various resources by our own work and have been duly acknowledged in the reference part.

In addition, other comments, reviews and data used by other authors, and organizations have been acknowledged, and explicitly cited.

We will take full responsibility for any fraud detected in my thesis. Ton Duc Thang University is unrelated to any copyright infringement caused on ourwork (if any).

Ho Chi Minh city, 22th August 2024

Author

(Signature and full name)

Lieu Dang Khoa

Nguyen Minh Chi

XÂY DỰNG MÔ HÌNH HỎI-ĐÁP DỰA TRÊN RAG CHO BỘ LUẬT DÂN SỰ VIỆT NAM

TÓM TẮT

Dự án này nhằm tạo ra một mô hình Hỏi-Đáp (QA) dựa trên RAG (Retrieval-Augmented Generation) được thiết kế để trả lời các câu hỏi liên quan đến Bộ luật Dân sự Việt Nam. Phương pháp nghiên cứu bao gồm thu thập dữ liệu, đề xuất mô hình, và phát triển hệ thống đối thoại tự động. Giai đoạn ban đầu bao gồm việc thu thập các tài liệu pháp lý và các cặp câu hỏi-đáp liên quan, sau đó thu thập một tập hợp bộ dataset gồm các loại văn bản pháp luật khá nhau. Các giai đoạn tiếp theo tập trung vào việc tinh chỉnh các mô hình ngôn ngữ lớn cũng như xây dựng hệ thống RAG có khả năng hiểu và tạo ra các cuộc đối thoại pháp lý. Dự án sẽ hoàn thiện bằng việc phát triển một hệ thống đối thoại có giao diện thiện với người dùng dựa trên RAG, sử dụng tập hợp văn bản và mô hình ngôn ngữ lớn để tạo điều kiện truy cập thông tin pháp lý một cách trực quan và khuyến khích sự tương tác của người dùng với Bộ luật Dân sự. Cách tiếp cận này kết hợp các phương pháp từ tin học pháp lý, học máy, và xử lý ngôn ngữ tự nhiên để giải quyết các thách thức đặc thù trong việc hiểu và tương tác với văn bản pháp luật.

BUILDING A RAG BASED QUESTION–ANSWERING MODEL FOR VIETNAM CIVIL CODE

ABSTRACT

This project aims to create a RAG (Retrieval-Augmented Generation)-based Question-Answering (QA) model tailored for answering questions related to the Vietnam Civil Code. The research methodology encompasses data collection, model suggestion, and the development of an automated conversational system. The initial phase involves gathering legal documents and relevant question-answer pairs,

followed by the gathering of a specialized corpus derived from this data. The next set of phases focuses on fine-tuning Large Language Models as well as building RAG pipeline that can understand and produce legal conversations. The project culminates in the development of an RAG-based automated conversational user-friendly system that uses the corpus and large language model to enable intuitive access to legal information and encourage user interaction with the Civil Code. This multidisciplinary strategy combines methods from legal informatics, machine learning, and natural language processing to address the unique challenges inherent in legal text comprehension and interaction.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AI	Artificial Intelligence
API	Application Programming Interface
BLEU	Bilingual Evaluation Understudy Score
CSS	Cascading Style Sheet
GPT	Generative Pre-trained Transformer
HTML	HyperText Markup Language
BLEU	Bilingual Evaluation Understudy Score
JS	JavaScript
LLM	Large Language Model
ML	Machine Learning
NLP	Natural Language Processing
PIP	Pip Installs Packages
RAG	Retrieval-Augmented Generation
URL	Uniform Resource Locator

CHAPTER 1 INTRODUCTION

1.1 Background

Vietnam's legal landscape centers around its Civil Code, which forms the backbone of the country's legal framework. Covering everything from property rights to family relationships, this extensive legislation governs civil life for individuals, organizations, and institutions. As Vietnam rapidly develops through industrialization and globalization, its legal system adapts to new challenges to ensure smooth operations.

In recent years, Vietnam experienced rapid socio-economic development, marked by industrialization, urbanization, and globalization. To keep up with these changes, the legal system is adapting to new challenges and making sure everything runs smoothly.

Although the Civil Code plays a vital role in establishing legal standards and controlling social relations, it can be difficult for individuals, corporations, legal professionals, and legislators to access and understand its contents. The complexity of legal language, coupled with the sheer volume of legal texts, presents barriers to understanding and utilizing the Civil Code effectively.

Moreover, the traditional methods of legal inquiry, such as consulting legal experts or reading physical copies of legal documents, may be time-consuming, costly, and inaccessible to many, especially those who are not well-versed in law practices. As a result, the development of a tailored Question-Answering (QA) model for the Vietnam Civil Code can be a life saver for many people. By harnessing the power of computational linguistics and machine learning, the model could make legal information more accessible to everyone. It would help people understand the Civil Code better, encouraging more people to learn about the law and get involved in legal matters in Vietnam. In this specific project, we will utilize the Retrieval-Augmented Generation (RAG) method to build a QA model.

In the following sections, this thesis outlines the objectives, research methodology, and structure of the proposed QA.

1.2 Research background

Vietnam's legal system is often perceived as complex, lengthy, and inaccessible to many due to its difficult legal language and cumbersome processes that can exhausting to understand. It might be challenging to obtain legal information and interpret the Vietnam Civil Code and all of its nuances, particularly for people who are not familiar with the legal system, businesses, or even legal professionals. The traditional methods of legal inquiry, such as consulting experts or reading extensive legal texts, are often time-consuming and costly.

This project aims to tackle these challenges by enhancing accessibility to the Vietnam Civil Code. By leveraging advanced computational linguistics and machine learning techniques, the project seeks to simplify the process of accessing legal information. We hope that a creation of a tailored Question-Answering model based on the Retrieval-Augmented Generation (RAG) method would facilitate more natural interactions between users and legal documents. This approach not only aims to make legal practices more accessible but also empower individuals to understand legal knowledge easier.

1.3 Research goals

Enhancing Accessibility:

The development of a RAG-based question-answering model for the Vietnam Civil Code aims to significantly enhance accessibility to legal information. This project aims to increase accessibility for a wide range of users, including consumers, corporations, and legal professionals, to the extensive and frequently difficult legal knowledge found in the Civil Code. Historically, legal documents required specialized knowledge to interpret and navigate, frequently requiring the assistance of legal specialists. This project seeks to make access to legal information easier by utilizing AI technology, enabling even individuals without professional legal training to receive precise and understandable answers to their legal enquiries. This method not only gives people the knowledge they need to know their legal rights and responsibilities, but it also makes dealing with the legal system.

Streamlining Legal Inquiry:

One of the primary objectives of this project is to streamline the process of legal inquiry by offering a user-friendly and intuitive interface to interact with the Vietnam Civil Code. The complexity of legal language often presents a barrier to effective communication, particularly for laypersons who may struggle to articulate their questions in legal terms. In order to solve this problem, the system allow users ask to questions in simple, everyday language, and the AI model interprets them to deliver contextually rich and accurate legal solutions.

Advancing Legal Informatics:

This project also represents a significant contribution to the field of legal informatics, an interdisciplinary domain that explores the application of information technology to legal problems. By developing innovative approaches to the comprehension and interaction with legal texts, this project bridges the gap between legal theory and computational linguistics. It pushes the limits of legal technology by examining how sophisticated AI models, like RAG, may be used to analyze and comprehend complex legal documents. In addition to improving the state of legal informatics today, this investigation establishes the foundation for next studies and advancements in this rapidly developing field.

1.4 Research methodology

This study adopts a comprehensive approach integrating theoretical foundations with practical application:

Theoretical Framework: Analyzing legal informatics, Question-Answering models, and existing RAG-related papers.

Practical Implementation: Developing a tailored QA model using the Retrieval-Augmented Generation (RAG) method, which combines the retrieval of relevant documents with advanced generation techniques to create accurate answers.

CHAPTER 2 THEORETICAL BASIS

2.1 Large Language Model

2.1.1 Overview

Large Language Models (LLMs) have become a cornerstone of modern natural language processing (NLP) and artificial intelligence (AI). These models are essential for tasks like text summarization, question answering, and machine translation since they are built to comprehend and produce human language with a high degree of accuracy. This section will examine the fundamental ideas, structures, and developments in LLMs that render them appropriate for use in the construction of a RAG-based question-answering model.

2.1.2 Foundations of Large Language Models

The foundation of LLMs is deep learning, notably the use of neural networks trained on huge amount of textual data. The main breakthrough in LLMs is their capacity to learn high-dimensional representations of words, phrases, and even complete texts, therefore capturing complex linguistic patterns and relationships.

LLMs are primarily composed of neural networks, especially those that use the Transformer architecture. The Transformer model, which was presented by Vaswani et al. in 2017, leverages a mechanism known as self-attention to capture dependencies between words regardless of the distance between them in a phrase. Compared to earlier models such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs), this method enables the model to comprehend context more efficiently.

Self-Attention Mechanism: By attending to different parts of the input sequence, the model is able to evaluate the importance of each word to each other, allowing it to build a rich, contextual representation of the text.

Positional Encoding: Transformers process the full sequence at once, in contrast to RNNs, which process data sequentially. Positional encoding is used to provide information about the position of words within the sequence, which is crucial for understanding the order and structure of the text.

2.1.3 Relevance of LLMs in question-answering task

2.1.3.1 Overview

The capabilities of LLMs make them particularly well-suited for question-answering tasks, where the model must understand a query and generate a relevant and accurate response. Accurately understanding legal writings, like the Vietnam Civil Code, requires LLMs to be able to comprehend context and subtle nuances.

2.1.3.2 Contextual Understanding and Response Generation

Contextual comprehension is a critical skill for legal question answering, and LLMs excel at it. These models are able to produce responses that are both legally correct as well as appropriate by utilizing the contextual representations that were acquired throughout training.

Handling Ambiguity: Legal writings frequently have wording that is unclear and needs to be carefully interpreted. LLMs, particularly those refined on legal corpora, can assist in clarifying terminology and offering accurate responses contingent upon the situation.

Scalability: LLMs can be scaled to handle large amounts of legal data, making them capable of answering complex questions that require the integration of multiple external sources of information.

2.1.3.3 Integration with Retrieval-Augmented Generation (RAG)

Retrieval mechanisms are combined with the advantages of LLMs in RAG models to improve the relevance and accuracy of generated answers. RAG-based models are able to produce more accurate and well-informed answers to legal queries by pulling relevant texts from a legal database and using an LLM to construct responses.

Enhanced Answer Accuracy: Retrieval mechanisms are used to assist in retrieving the most relevant legal materials, which the LLM can use to produce a precise response to the legal query. This method is especially helpful in the field of legal informatics, where accuracy of the data is crucial.

Adaptability to Legal Domains: RAG models are able to be adjusted to particular legal domains, such the Vietnam Civil Code, guaranteeing that the generated responses are both legally sound and contextually appropriate.

2.2 Vietnam’s Legal System and Legal Informatics

2.2.1 Introduction to Vietnam’s Civil Code

The Civil Code of Vietnam is a heart of the country’s legal framework, governing civil life for individuals, organizations, and institutions. The Civil Code provides a thorough legal framework for civil transactions and interactions, and it is crucial for defining legal standards and regulating social connections.

However, there are obstacles to comprehending and properly applying the Civil Code because of its intricacy and the legal terminology employed. For many people, especially those who are not familiar with legal procedures, the traditional techniques of legal inquiry—such as speaking with legal experts or reviewing hard copies of court documents—can be expensive, time-consuming, and inaccessible. This emphasizes how beneficial it is to use technology to enhance legal information access and comprehension, which is a major area of study for legal informatics.

2.2.2 Introduction to Legal Informatics

Technology can be used to improve legal processes and practices, legal informatics seeks to address issues facing the legal profession and raise the standard of legal services, research, and education while also increasing efficiency and accessibility.

At its core, legal informatics focuses on several key objectives. First, it involves developing computational techniques to analyze diverse legal texts such as regulations, case law, and contracts. These techniques help extract meaningful insights and patterns from legal documents, aiding in decision-making processes.

Secondly, legal informatics aims to create structured models that represent legal knowledge in a format understandable by computers. Legal information retrieval and manipulation are made easier by this structured representation, which helps with tasks like legal research and analysis.

Moreover, the field designs algorithms and systems to retrieve relevant legal information from extensive databases efficiently. By automating information retrieval processes, legal informatics can considerably increase the speed and accuracy of accessing critical legal data.

Additionally, legal informatics involves building expert systems that emulate the reasoning and decision-making capabilities of legal professionals. These systems improve the competence of legal practitioners by helping with activities like evaluating difficult legal cases that are have nuances and complexities and giving legal counsel.

Lastly, legal informatics promotes the development of electronic publishing platforms and standards. By facilitating the spread of legal information, these platforms improve public, legal professional, and researcher accessibility.

2.3 Question-answering task

2.3.1 Overview

Question-Answering (QA) is a field within natural language processing (NLP) that focuses on building models that are capable of providing answers to questions in natural language. These systems interpret the query, search relevant data sources, and provide accurate and concise answers. QA systems are a fundamental area of natural language processing (NLP).

2.3.2 Key features

Natural Language Understanding:

QA systems must understand and comprehend questions in natural language. This involves identifying the main components (subject, verb, and object) as well as understanding the main context behind the question.

Information Retrieval:

QA systems must be able to search through large datasets or corpora to find relevant information to the question. This can be done by using algorithms to retrieve documents, passages or snippets that are likely to contain the answer to the main question.

Answer Extraction:

After locating the relevant information, the QA model must extract the answer precisely. The model then must generate an answer based on the retrieved information.

2.4 Overview of RAG (Retrieval augmented generation)

2.4.1 The limitation with LLMs

While LLMs have revolutionized the field of natural language processing with remarkable capabilities in generating text, they have several limitations:

Firstly, LLMs often struggle with providing accurate and up-to-date information. Since these models can only generate text based on the information gathered from past vast amount of data that last only lasted a specific duration (like from 2016 – 2021). As such, when asked about current occurrences or recently discovered facts, they could respond with information that is out-of-date or inaccurate. An interesting example would be ChatGPT 3.5, despite being one of the most popular chatbots, it can only generate responses based on the training data it was provided with up until its cutoff date, which is 2021. This means it lacks access to real-time information, which can lead to outdated or inaccurate answers regarding recent events or new developments.

Second, it sometimes hallucinates by generating plausible but incorrect or nonsensical answers due to its probabilistic nature and the absence of grounding in external, up-to-date information sources. This problem stems from the model's heavy reliance on existing outdated training data as opposed to publicly available data. This limitation is particularly problematic in applications requiring high levels of accuracy and reliability, such as legal, where incorrect information can lead to serious consequences.

2.4.2 Introduction to RAG

Retrieval-Augmented Generation (RAG) represents a significant advancement in the field of natural language processing, specifically in enhancing the capabilities of large language models (LLMs). Even though LLMs are incredibly good at NLP tasks, they have drawbacks such the possibility of producing text that is factually wrong and a limited capacity to refer to specific or current information. RAG tackles these issues by integrating external information into the language creation process

RAG is a hybrid strategy that combines the advantages of retrieval-based and generation-based models. The core idea behind RAG is to enhance the generation capabilities of models by integrating a retrieval mechanism that fetches relevant documents or passages from a large corpus, which can then be used to generate more accurate and contextually appropriate answers.

At its core, RAG has two main components: retrieval and generation. During the retrieval phase, relevant document chunks are extracted from a vast external knowledge base through semantic similarity calculations. This guarantees that the generated replies are based on true information, thereby reducing the problem of hallucinations that frequently afflict conventional LLMs. By integrating external knowledge, RAG can provide more accurate and contextually appropriate answers.

2.4.3 Components of RAG

The RAG models consist of these primary components:

Retriever: The retriever is responsible for identifying and fetching relevant documents and passages from a large corpus or dataset based on the input query. Several popular retriever models have been on the rise recently. Those models can efficiently retrieve relevant documents or passages:

- **Dense Passage Retrieval:** Dense Passage Retrieval (DPR) is a state-of-the-art retrieval model developed by Facebook AI. It uses dense vector representations for both queries and documents. The vectors are generated using pre-trained language models like BERT.
- **BM25 (Okapi BM25):** BM25 is a probabilistic information retrieval model that uses a formula that takes word frequency and document length into account to rank documents according to how relevant they are to the query.

Generator: The generator component is responsible for generating coherent, well-meaning and contextually relevant answers based on the input queries and the retrieved documents. This component utilizes advanced sequence-to-sequence transformer models to generate high-quality natural language text, integrating information from retrieved passages to ensure accuracy and relevance.

The generator relies on sequence-on-sequence models to function. These models are designed to map an input sequence, which includes the query and the retrieved passages, to an output sequence, which is the generated response. Among the popular Seq2Seq models are BART (Bidirectional and Auto-Regressive Transformers) and T5 (Text-to-Text Transfer Transformer).

The generator merges context from retrieved documents with the input query to produce a comprehensive response. Using attention mechanisms and an encoder-decoder architecture, it focuses on relevant parts of the documents. The process involves fusing information from multiple passages to generate coherent and accurate responses, followed by refining the output through post-processing.

2.4.4 Retrieval

The retrieval step plays a crucial role in enhancing the capabilities of LLMs by referencing external sources. There are several key features of retrieval worth discussing.

2.4.4.1 Retrieval source

RAG systems rely on diverse external knowledge sources to power LLMs. Initially focused on unstructured text, retrieval sources have expanded to include semi-structured data (such as PDFs) and structured data from knowledge graphs (KGs). Using content generated by LLMs for retrieval has also become a popular approach, with the goal of utilizing the internal knowledge of the model to improve performance in providing accurate responses.

2.4.4.2 Retrieval granularity

The granularity of retrieval units impacts the relevance and efficiency of RAG systems. Options range from fine-grained (tokens, phrases, sentences) to coarse-grained (documents, knowledge graphs). While fine-grained units can provide precise information, they may increase computational costs and introduce noise. Coarse-grained units, on the other hand, may overlook specific details but streamline the retrieval process.

2.4.4.3 Indexing optimization

Effective indexing is critical for optimizing the retrieval phase in RAG systems. Options range from chunking documents into manageable segments, enriching chunks with metadata (e.g., timestamps, document structure), and constructing hierarchical structures for efficient traversal and retrieval. By using these techniques, information extracted during the process can be more accurately.

2.4.4.4 Query optimization

Techniques include query expansion to provide additional context, query transformation to refine original queries, and query routing based on semantic or metadata-driven approaches. These methods ensure that RAG systems retrieve the most relevant information aligned with user queries, enhancing overall performance.

2.4.5 Generation

2.4.5.1 Context Curation

Retrieving and directly inputting all retrieved information to a Large Language Model (LLM) for answering questions is not recommended. Adjustments should be made from two perspectives: adjusting the retrieved content and adjusting the LLM.

2.4.5.2 Reranking

Reranking reduces the total document pool by rearranging document chunks to emphasize the most relevant results first. By serving as both an enhancer and a filter, this improves information retrieval and provides cleaner inputs for more accurate language model processing. Reranking can be performed using rule-based methods based on predefined metrics like Diversity, Relevance, and Mean Reciprocal Rank (MRR), or model-based approaches such as Encoder-Decoder models from the BERT series, specialized reranking models like Cohere rerank or bge-reranker-large, and general large language models like GPT.

2.4.5.3 Context Selection/Compression

A common misconception in the Retrieval-Augmented Generation (RAG) process is that retrieving as many relevant documents as possible and concatenating them to form a lengthy retrieval prompt is beneficial. However, too much context could introduce more noise and hinder the LLM's ability to recognize important details. Certain methods detect and eliminate irrelevant tokens using small language models (SLMs) such as LLaMA-7B or GPT-2 Small, which changes the context into a format that is difficult for humans to understand but easy for LLMs to understand. Other techniques involve employing contrastive learning to train an information extractor or information condenser in order to efficiently compress context.

In addition to compressing the context, reducing the number of documents helps improve the accuracy of the model’s answers. For instance, the “Filter-Reranker” paradigm combines the strengths of LLMs and SLMs, with SLMs serving as filters and LLMs functioning as reordering agents. Another approach involves having the LLM evaluate the retrieved content before generating the final answer, allowing the LLM to filter out documents with poor relevance through LLM critique.

2.4.6 Working mechanism of RAG

The RAG model operates in two main stages:

Retrieval stage: When a question is presented, the retriever searches through datasets or corpora to find passages that contain relevant information to the question.

Generation stage: The retrieved passages, along with the original question, are then fed into the generator. The generator processes the information from the passages and produce a concise and comprehensive answer. This allows the model to generate accurate and context-rich answers.

The inner working of RAG will be discussed below in chapter 3.

2.4.7 Applications

The RAG method has been utilized in several applications:

- **Open-Domain Question Answering**: Providing answers to general knowledge questions by retrieving and synthesizing information from vast datasets like Wikipedia and social media. Examples: Google Gemini and ChatGPT 4.
- **Customer Support**: RAG-based systems can help provide more accurate and contextually relevant responses by retrieving information from product manuals or support documents.
- **Legal Document Analysis**: Assisting legal professionals such as lawyers by answering complex legal questions based on retrieval from legal texts and case law databases.
- **AI-assisting text summarization**: RAG can help generate precise and contextually relevant summaries of large text documents, saving users time to search for answers they need.

2.4.8 Advantages and disadvantages of RAG

RAG offers several advantages and disadvantages to NLP-related tasks:

Advantages:

- **Improved Accuracy**: By retrieving relevant passages from a large corpus or dataset, RAG models can provide more accurate and detailed answers compared to purely generative models that rely solely on their internal knowledge.
- **Contextual Relevance**: The retrieval component makes sure that the generated replies are based on factual data, which enhances the contextual relevance and reliability of the responses.
- **Scalability**: RAG models are designed to handle large-scale corpora, making them suitable for a wide range of applications, from open-domain question answering to specialized fields such as legal document analysis.

Disadvantages:

- **Reliance on retrieval component:** The effective of RAG-based models rely heavily on the accuracy of the retrieval component. The RAG can only generate high-quality answers if the most relevant documents are retrieved.
- **Computation and memory**: RAG models require significant computational resources and memory, especially if the dataset is considerably large in size.
- **Application adaptation**: RAG models have to be fine-tuned to suit specific needs and serve specialized domains, such as customer support or law assistant chatbot.
- **Prone to bias:** The model may inherit biases present in the training data and the retrieval corpus, leading to biased or unfair responses. Addressing these biases is a non-trivial task.

2.5 RAG variants

Recent advancements in RAG have seen the developments of various alternative approaches that enhance the accuracy, relevance and reliability of generated responses. During our research, we have explored and looked into several notable variants of RAG including Corrective Retrieval Augmented Generation (CRAG), Self-RAG, RAG-Fusion, and Query Rewriting for Retrieval-Augmented Large Language Models.

2.5.1 Corrective Retrieval Augmented Generation (CRAG)

CRAG operates in several key stages to enhance the effectiveness of RAG systems. A large language model (LLM) generate text based on an initial prompt or query. Subsequently, it obtains necessary documents or papers from relevant sources. The defining feature of CRAG is the retrieval evaluator, which assess the retrieved documents for their relevance. This assessment phase ensures that the information is appropriate to be used for supporting or enhancing the generated response.

Based on the evaluation, CRAG can employ various actions to refine the response:

- Correcting factual inconsistencies that may exist in the generated text.
- Removing irrelevant or misleading information sourced from the retrieved documents.
- Prompting additional document retrieval to supplement existing information or validate key points within the generated response.

By integrating text generation with information retrieval and evaluation, CRAG enhances the overall precision and relevance of responses. This approach not only improves the efficiency of RAG systems but also enhances their ability to deliver informed and contextually appropriate answers across various applications.

2.5.2 Self-RAG

Self-Retrieval Augmented Generation (Self-RAG) is an innovative variant of the RAG systems. In addition to leveraging the existing functionalities of RAG itself, it also integrates a layer of self-reflection, allowing the model to evaluate itself by examining the quality of the retrieved documents.

Self-RAG operates mainly by self-reflection during both retrieval and generation stages:

Self-RAG begins with a large language model (LLM) generating text based on an initial prompt or query. It then incorporates self-reflection tokens within the retrieval process. These tokens enable the model to introspectively evaluate the quality of retrieved documents by asking questions such as:

- “Is the retrieved information relevant?”
- “Are these documents from reliable sources?”

Based on this assessment, self-RAG adjusts its retrieval strategy:

- If the reliability is questionable, it might prioritize documents from trusted sources.
- If the retrieved documents are deemed irrelevant, it could refine the search query.

This self-reflective ability allows Self-RAG to automatically enhance the accuracy and contextual relevance of the generated answers. By dynamically adjusting to the quality of the retrieved information, Self-RAG ensures that the final output answer is refined.

2.5.3 RAG-Fusion

RAG-Fusion presents significant advancement in Retrieval-Augmented Generation (RAG) techniques by enhancing the retrieval process and response stages. Unlike traditional RAG systems that only employ a single query, RAG-Fusion employes multiple similar queries generated from the original prompt. By doing this, the retriever can retrieve more relevant documents and thus improving the effectiveness of generated responses.

1. Initial prompt: RAG-Fusion begins with receiving the initial prompt provided by the user. This prompt will be used to generate queries.

2. Queries Generation: Instead of relying on a single query, RAG-Fusion will generate multiple queries which are similar to the original query. For example, from the original query of “Is it illegal to run a red light?”, multiple similar queries such as “Will you go to jail if you run a red light?”, “What is the maximum fine if you get caught running a red light?”. Afterwards, with the k number of queries, the retriever will receive k number of documents.
3. Re-ranking: The main feature of RAG-fusion is its ability rank relevant documents and only keep the most relevant ones. RRF analyzes the ranking of documents across multiple queries and combines them into a unified ranking that prioritizes the most relevant documents at the top.
4. Generation: Based on the ranking, RAG-Fusion combines the information retrieved from multiple documents and their corresponding queries to generate a comprehensive answer to the prompt. This ensures that the generate response is not only accurate but also rich in multiple contex

CHAPTER 3 PROPOSED MODEL

3.1 Data collection

3.1.1 Overview

In our project, the primary focus is on the Vietnam Civil Code, which serves as the foundation for the RAG-based question-answering bot. The legal documents used are sourced mainly from thuvienphapluat.vn and luatvietnam.vn, which are reputable repositories for Vietnamese legal information. This source provides comprehensive and up-to-date legal texts, ensuring that the data used in our chatbot is both accurate and relevant.

3.1.2 Data for question-answering LLMs

For fine-tuning our question-answering LLM, we compiled a dataset comprising various question-answer pairs specifically related to the Vietnam Civil Code. This data is crucial for enabling the chatbot to provide accurate and contextually relevant responses in the legal domain.

3.1.2.1 Data gathering process

To gather as many question-and-answering pairs as possible, we created a script that automates the extraction of question-and-answer pairs and saves them to a file. It operates by sending requests to multiple pages on the website, parsing the HTML content, and identifying questions and their corresponding answers. The script uses specific HTML tags (h2 for multiple questions and strong for single questions) to locate these elements. Once the questions and answers are identified, the script cleans up any unnecessary HTML tags and stores the extracted information in a JSON lines file. It loops through a range of pages, processing each page's content to gather the desired data.

Key technologies used:

- **Python**: Used for scripting the data scraping and processing tasks.
- **BeautifulSoup**: A Python library used to parse HTML and extract necessary data from web pages.

- **Requests**: A Python library utilized to send HTTP requests and retrieve the HTML content from web pages.
- **JSON**: Used for storing the extracted question-answer pairs in a structured format, enabling easy access and processing during model fine-tuning.

3.1.2.2 Data statistics

Table 3-1: QA dataset statistics

Statistics	**Value**
Source	Thuvienphapluat.vn
Number of Rows	35181
Average Question Length	74.608823
Average Answer Length	318.148518
Most Common Words	[(có, 248734), (của, 219278), (định, 189396), ...

3.1.3 Legal documents

In addition to scraping question-answer pairs, our project involves downloading legal document files. These documents are essential for building a comprehensive legal knowledge base for our chatbot model to search for relevant information and then fetch answers. The following steps outline the process for downloading legal documents.

3.1.3.1 Data gathering process

We created a script which works by visiting specific web pages, searching for links to downloadable files, and saving those files to the user’s computer. The script systematically goes through multiple pages of the website, identifies links to both PDF and Word documents, and then downloads them.

The main steps include constructing the URLs for each page, identifying the download links on those pages, and then saving the documents locally. It keeps track of the number of files downloaded and provides feedback on the process. This automation allows for efficient bulk downloading of documents from the website.

Key technologies used:

- **Python**: For scripting and executing the program.
- **Requests**: For sending HTTP requests to fetch web pages.
- **BeautifulSoup**: For parsing HTML content to extract data.
- **OS**: For file handling operations.

3.1.3.2 Data statistics

Table 3-2: Legal documents statistics

Statistics	**Value**
Source	Thuvienphapluat.vn, luatvietnam.vn
Number of files	386
File type	PDF
Legal document types	Decrees, Circulars, Civil Code 2015, Resolutions

3.2 Fine-tune an LLM for Vietnamese Legal question-answering

3.2.1 Choose a suitable LLM

For this task, we explore the application of state-of-the-art (SOTA) Large Language Models (LLMs) for the task of Vietnamese legal question-answering. The objective is to evaluate the effectiveness of these models in handling Vietnamese text and providing accurate responses to legal queries.

We have chosen three SOTA LLMs working well with Vietnamese text, including:

- PhoGPT: a 4B parameter model specifically trained on Vietnamese data, making it well-suited for Vietnamese language understanding.
- Vistral: a 7B parameter model known for its strong performance on various Vietnamese language tasks, such as question answering, text summarization,..
- Vinallama: a 7B parameter model, also demonstrate strong performance on Vietnamese text tasks and exhibiting high accuracy in legal document analysis.

Reasons to choose these models can be explained by:

- Open-source: all models are freely available, allowing for free access and experiment.
- State-of-the-art-performance: Each model has consistently demonstrated excellent performance in various Vietnamese language tasks, signifying their ability to handle complex legal language.
- Vietnamese Text proficiency: These models have been specifically trained on Vietnamese data, enabling them to deeply understand the nuances of the language, including legal terminology, and handle Vietnamese-specific legal documents.

This research aims to contribute to the advancement of Vietnamese legal technology by exploring the potential of these SOTA LLMs for legal question-answering. By leveraging the strengths of these models and addressing their limitations through rigorous evaluation and fine-tuning, we can pave the way for efficient and accessible legal assistance for the Vietnamese population.

3.2.2 Fine-tune with QLoRA

3.2.2.1 Difficulties when fine-tuning LLMs

When fine-tuning LLMs, we encounter some difficulties. For storing 1B parameters with fully accurate 32-bit float-point numbers require approximately 4BG of storage space. This is because each parameter occupies 4 bytes (32 bits).

Table 3-3: Memory Requirements for Fine-Tuning LLMs with 32-bit Floating-Point Precision

	Bytes per parameter
Model parameters (weights)	4 bytes per parameter
Adam optimizer (2 states)	+8bytes per parameter
Gradients	+4 bytes per parameter
Activation and tem memory (variable size)	+4 bytes per parameters

Total	**= 4 bytes per parameter + 20 extra bytes per parameter**

In addition to parameter storage, fine-tuning requires extra memory for:

- Activation Functions: These activations, such as ReLU and Softmax, are used during forward pass and need to store intermediate activations.
- Gradient: the core process of fine-tuning, relies on calculating of each parameter. These gradients require additional storage.
- Other Buffers: Various temporary buffers are need for efficient computation, further increasing memory demand.

Combining these model parameters and the memory needed for fine-tuning, a rough estimate suggests a total memory requirement of around 4GB to fine-tuning a 1GB-parameter model.

Scaling up to **4B-parameter model**, it implies a storage requirement of around 16GB and a total fine-tuning memory demand close to **24GB**.

Similarly, with **7B-parameter models**, it would require around **28GB** for storing model parameters and approximate **42GB** for fine-tuning.

This raises a significant difficulty for us as we just rely on free GPU T4 provided by Google Colab which is only 15GB. This means that fine-tuning model exceeding 1B parameters become problematic, as the memory requirements quickly exceed the available capacity. This constraint poses a serious challenge for researchers and developers working with LLMs.

To address this, we employ a technique called Parameter Efficient Fine-Tuning (PEFT), specifically in this task is QloRA.

3.2.2.2 QloRA

To overcome the memory constraints of fine-tuning large LLMs, we employed a technique called Parameter-Efficient Fine-Tuning (PEFT), specifically qLoRA (Quantized Low-Rank Adaptation).

qLoRA is a powerful and efficient method for fine-tuning LLMs while significantly reducing memory requirements. It works by introducing a low-rank adaptation to the model's weights, allowing for fine-tuning with minimal changes to the original model parameters.

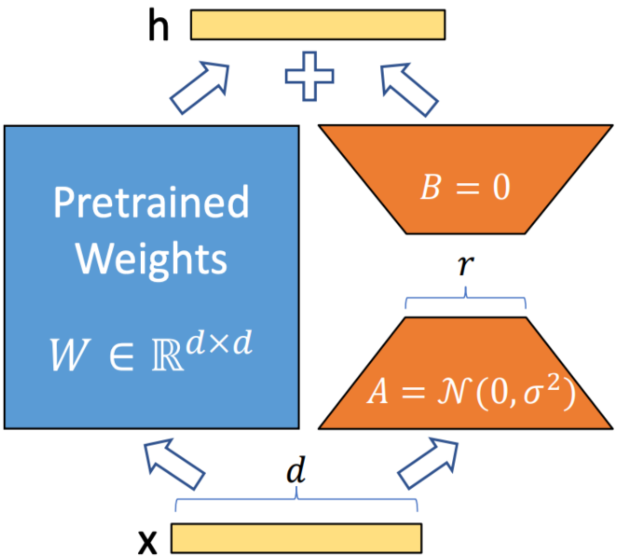

Figure 3-1: Overview of QLoRA (Quantized Low-Rank Adaptation)[1]

The qLoRA works as follow:

1. Low-Rank Decomposition: The model's weights are decomposed into low-rank matrices. This decomposition allows for representing the weights with a significantly smaller number of parameters.
2. Parameter-Efficient Adaptation: Only the low-rank matrices are fine-tuned, while the original model weights remain frozen. This significantly reduces the number of parameters that need to be updated, minimizing memory requirements and computational cost.
3. Quantization: The low-rank matrices are quantized to 4-bit precision, further reducing memory usage and improving inference efficiency.

3.3 Retrieval-augmented Generation

3.3.1 Pre-retrieval phase

[1]https://medium.com/@ashkangolgoon/understanding-qlora-lora-fine-tuning-of-llms-65d40316a69b

To enable efficient and accurate retrieval of relevant legal information during the RAG process, a crucial step involved generating vector embeddings for the Vietnamese Civil Code documents and storing them in a remote vector database. This pre-retrieval phase ensured that the legal knowledge base was readily accessible for semantic search and retrieval.

The Vietnamese Civil Code documents, in PDF format, were first parsed into a structured representation suitable for embedding and indexing.

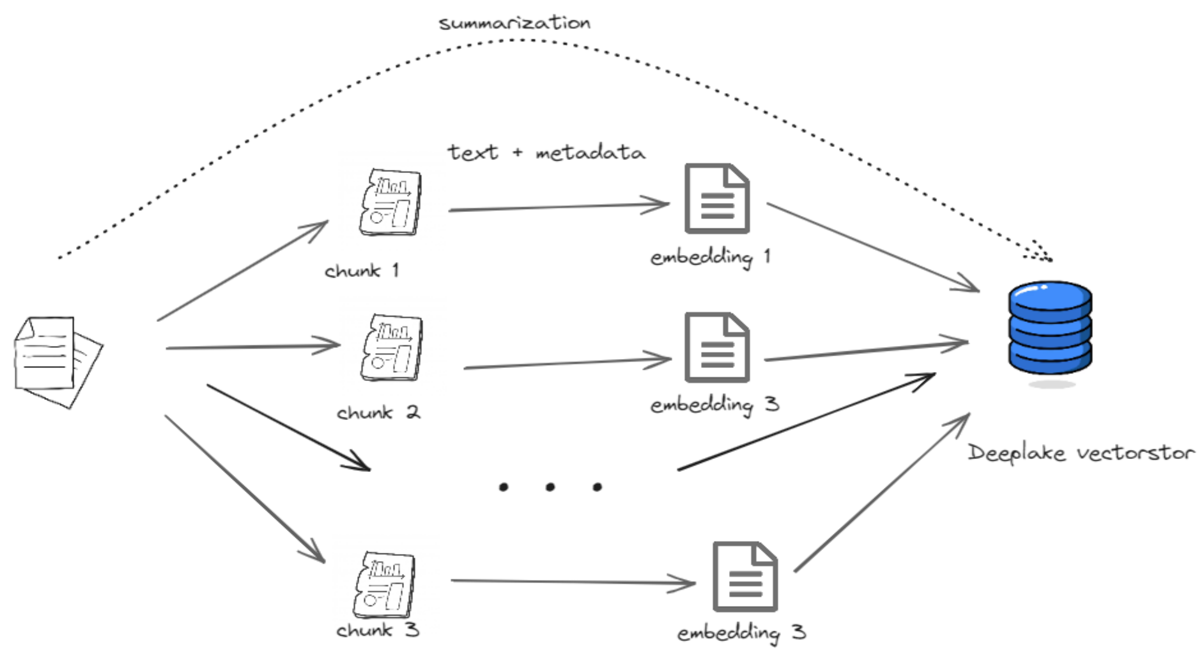

Figure 3-2: Embedding documents

To generate vector embeddings that capture the semantic meaning of the legal text, the *BAAI/bge-small-en-v1.5* embedding model was selected. This model was chosen due to its versatility in handling multi-functional, multi-lingual, and multi-granularity tasks.

To optimize the retrieval process and enhance the contextual understanding of the legal information, two chunking strategies were employed:

Sentence Window Node Parser: This approach split a single sentence into a chunk, accompanied by a "window context" containing the five preceding and five succeeding sentences. This method proved particularly useful for large documents, facilitating retrieval of fine-grained details.

Semantic Chunking: This strategy dynamically identified chunk boundaries between sentences based on embedding similarity, ensuring that chunks contained semantically related sentences. This approach minimized the risk of dividing conceptually connected text into separate chunks.

The generated embeddings, along with their associated chunks, were then loaded into a remote DeepLake vector database for efficient retrieval.

By meticulously preparing the vector embeddings, employing appropriate chunking strategies, and leveraging the capabilities of a remote vector database, we established a robust and efficient retrieval infrastructure for the RAG system. This ensured that the chatbot could quickly access and process the relevant legal information during the retrieval stage, enhancing its ability to provide accurate and informative legal advice to Vietnamese users.

3.3.2 Retrieval phase

To address the inherent ambiguity and potential lack of clarity in user queries, particularly in the legal domain, a query expansion step was implemented prior to retrieval. This step aimed to generate multiple variations of the original query, covering different aspects of the user's intent. By broadening the search scope, the retrieval process was enhanced, ensuring that a wider range of relevant documents could be identified.

The query expansion process involved generating *n* (in this task is 5) related queries based on the user's initial input. This was achieved using the Gemini-1.5-Flask API, a powerful language model developed by Google AI. The Gemini model was leveraged to generate a series of refined queries, each focusing on a specific aspect of the user's intent.

To guide the Gemini model effectively, the chain-of-thought (CoT) prompting technique was employed. This technique encourages the model to provide a step-by-step reasoning process, leading to more comprehensive and relevant query expansions. The following prompt template was used:

```
gen_qa_prompt = """
Bạn là một trợ lý xuất sắc trong việc tạo ra các câu truy vấn tìm kiếm liên quan. Dựa trên câu truy vấn đầu vào dưới đây, hãy tạo ra {num_queries} truy vấn tìm kiếm liên quan, mỗi câu trên một dòng. Lưu ý, trả lời bằng tiếng Việt và chỉ trả về các truy vấn đã tạo ra.
Ví dụ:
# query: Muốn mở nhà hàng thì cần tuân thủ các quy định pháp lý nào?
# generated queries:
[- Những giấy tờ pháp lý nào cần thiết để mở một nhà hàng?,
- Quy trình xin giấy phép kinh doanh nhà hàng bao gồm những bước nào?,
- Những quy định về an toàn thực phẩm cần tuân thủ khi mở nhà hàng là gì?,
- Các điều kiện về vệ sinh môi trường đối với nhà hàng như thế nào?]

### Câu truy vấn đầu vào: {query}

### Các câu truy vấn:"""
```

Retrieve:

The retrieval phase plays a pivotal role in the RAG pipeline, extracting relevant information from the indexed legal documents based on the user's query. In this research, two approaches were implemented: Naive RAG and advanced RAG, each employing different retrieval strategies.

The Naive RAG approach utilized LlamaIndex's built-in query engine, relying primarily on a probabilistic retrieval method based on document similarity. While this approach provided a baseline for comparison, it often struggled to retrieve highly relevant documents, especially when faced with complex or ambiguous queries.

To overcome the limitations of the Naive approach, an advanced RAG strategy was implemented, employing a hybrid search method that combined the strengths of semantic similarity and keyword matching. This hybrid approach aimed to address the challenges posed by both vague and keyword-specific queries, ensuring comprehensive retrieval of relevant information.

The hybrid search method involved two distinct retrievers:

- Vector Retriever: This retriever leveraged the vector embeddings generated in the pre-retrieval phase, enabling semantic search based on document similarity. This approach proved effective in identifying documents with similar semantic content to the user's query, even if specific keywords were not present.
- BM25 Retriever: This retriever employed the BM25 algorithm, a traditional keyword-based retrieval method, to identify documents that contained specific keywords from the user's query. BM25 excels at matching queries with documents containing exact keyword matches, proving valuable for retrieving highly relevant information when the query is explicitly keyword-driven.

To ensure comprehensive retrieval, the following steps were taken:

- Combined Retrievers: Both the vector retriever and BM25 retriever were incorporated into a unified retrieval system.
- Query Expansion: The expanded set of queries (including the original query and its variations generated in the query expansion step) were used as input to both retrievers.
- Score Threshold: A score threshold was applied to filter out retrieved nodes with scores below a predefined value (e.g., 0.7). This threshold ensured that only highly relevant documents were considered for the final answer generation.

3.3.3 Post-retrieval

To enrich the contextual understanding of the retrieved documents, a metadata replacement post-processing step was applied. This step leveraged the "window context" information generated by the Sentence Window Node Parser during the pre-retrieval phase.

The Sentence Window Node Parser (described in section 4.3.1.3) appended a window of five preceding and five succeeding sentences to each retrieved node. This window provided crucial contextual information for the document chunk.

The MetadataReplacementPostProcessor class was used to replace the original text of the retrieved node with the "window context."

Reranking, a crucial step in the RAG pipeline, involved re-ranking the retrieved documents based on their relevance to the user's query. This step ensured that the most pertinent documents were selected for input to the LLM, leading to more precise and focused answers.

To perform reranking, the Cohere Reranker, a powerful multilingual model from Cohere AI, was used. This model leverages advanced language understanding capabilities to assess the semantic similarity between the user's query and the retrieved documents.

Ranking: The Cohere Reranker assessed the relevance of each retrieved document to the user's query and assigned a relevance score.

Top N Selection: The top *n* (in this task, 3) documents with the highest relevance scores were selected for input to the LLM.

3.3.4 Generation

After the retrieval and post-processing steps, the final stage of the RAG pipeline involved using a fine-tuned language model to generate a comprehensive and informative response to the user's query. To achieve efficient inference, we employed Ollama, an open-source, modular runtime for running large language models (LLMs).

Ollama was first installed from the official Ollama website, providing a lightweight and efficient runtime environment for LLMs.

The fine-tuned GGUF model file (Vistral-7B, fine-tuned using QLoRA), saved on Hugging Face, was downloaded to the local machine.

A configuration file, named "Modelfile," was created to specify the GGUF model file path, inference settings, and the prompt template.

```
FROM /teamspace/studios/this_studio/vistral-7b-legal-chat-final/unsloth.Q4_K_M.gguf

TEMPLATE """
Bạn là một chuyên viên tư vấn pháp luật tại Việt Nam với nhiều năm kinh nghiệm và kiến thức chuyên sâu. Nhiệm vụ của bạn là cung cấp câu trả lời và tư vấn pháp lý cho các câu hỏi của người dùng.

{{ if .Prompt }}
### Câu hỏi:
{{ .Prompt }}{{ end }}

### Trả lời:"""

PARAMETER num_keep 24
PARAMETER temperature 0.8
PARAMETER stop "<s>"
PARAMETER stop "</s>"
PARAMETER stop "<|im_start|>"
PARAMETER stop "<|im_end|>"
PARAMETER stop "### Câu hỏi:"
PARAMETER stop "### Trả lời:"
```

Explanation of Parameters:

- **FROM:** Specifies the path to the downloaded GGUF model file.
- **TEMPLATE:** Defines the prompt template, which includes:
- **Role-Playing Instructions:** The prompt introduces the LLM as a legal expert.
- **Question Input:** A placeholder for the user's query.
- **Answer Prompt:** A prompt that encourages the LLM to generate a comprehensive and informative answer.
- **PARAMETER:** Sets various inference settings:
- **num_keep:** The number of tokens to keep in memory during the generation process.

- **temperature:** A value that controls the randomness of the generated text. A higher temperature leads to more creative and unpredictable responses. We set it to 0.8 for a balance between randomness and coherence.
- **stop:** A list of tokens that, when encountered during generation, will signal the model to stop generating text. These tokens were chosen to ensure that the generated responses had a clear structure and did not include unnecessary information.

After configuring the "Modelfile," the following command was used to initialize the Ollama model: ***ollama create vistral-legal-chat -f Modelfile***

This command created a new Ollama model named "vistral-legal-chat," using the configurations specified in the "Modelfile."

When the model is intialized, we can use it directly in the command line with the command: ***ollama run vistral-legal-chat***

To inference the model in our RAG apps, we use the module Ollama supported by llama-index, for example:

The output is:

“Theo quy định hiện hành, luật Giao thông đường bộ 2013 là văn bản có giá trị cao nhất trong hệ thống các văn bản quy phạm pháp luật điều chỉnh hoạt động tham gia giao thông trên tất cả các loại hình phương tiện (ô tô, xe máy...) và người đi bộ. Luật được xây dựng dựa trên cơ sở Hiến pháp nước Cộng hòa xã hội chủ nghĩa Việt Nam 2013 và kế thừa những nội dung của hệ thống văn bản quy phạm trước đó là “Luật Giao thông đường bộ năm 2008” (hiệu lực đến hết ngày 01/01/2015).”

Following the retrieval and post-processing stages, the final step in the RAG pipeline involved crafting a contextualized prompt for the fine-tuned language model (LLM) to generate a comprehensive and informative response. This step aimed to leverage the retrieved legal context and guide the LLM to produce answers that were both human-like and legally accurate.

The top three documents retrieved after reranking were processed to extract relevant legal information and format it into a structured "context" for the LLM. This process involved:

Text Extraction: The text content from the top three documents was extracted, preserving the original formatting and structure.

Metadata Appending: To provide the LLM with additional context and ensure grounded legal responses, important metadata (including the name of the law and the page number where the information was retrieved) was appended to each document's text.

A meticulously crafted instruction prompt was designed to guide the LLM in generating the desired response. This prompt is as follow:

```
gen_rag_answer = """
Bạn là một trợ lý ảo về tư vấn pháp luật. Nhiệm vụ của bạn là sinh ra câu trả lời dựa vào hướng dẫn được cung cấp, kết hợp thông tin từ tài liệu tham khảo với khả năng suy luận và kiến thức chuyên môn của bạn để đưa ra câu trả lời sâu sắc và chi tiết.
Ví dụ: Nếu văn bản được truy xuất nói về một điểm pháp luật, nhưng câu hỏi liên quan đến một tình huống thực tế, bạn cần dựa vào thông tin đó để giải quyết hoặc trả lời thấu đáo câu hỏi.

### Quy tắc trả lời:
1. Kết hợp thông tin từ phần tài liệu tham khảo ## context với khả năng suy luận và kiến thức chuyên môn của bạn để đưa ra câu trả lời chi tiết và sâu sắc.
2. Trả lời như thể đây là kiến thức của bạn, không dùng các cụm từ như: "dựa vào thông tin bạn cung cấp", "dựa vào thông tin dưới đây", "dựa vào tài liệu tham khảo",...
3. Từ chối trả lời nếu câu hỏi chứa nội dung tiêu cực hoặc không lành mạnh.
4. Trả lời với giọng điệu tự nhiên và thoải mái như một chuyên gia thực sự.
### Định dạng câu trả lời:
1. Câu trả lời phải tự nhiên và không chứa các từ như: prompt templates, ## context...
2. Không cần lặp lại câu hỏi trong câu trả lời.
3. Trình bày câu trả lời theo format dễ đọc


----------------------
## context:
{context_str}
----------------------
```

```
## Câu hỏi:
{query_str}

## Trả lời:
"""
```

This process, incorporating contextualized prompting and answer generation, formed the core of the RAG pipeline, allowing the chatbot to provide accurate and user-friendly legal advice to Vietnamese users.

The final step involved providing the constructed prompt to the Ollama instance, which utilized the fine-tuned LLM (Vistral-7B) to generate a comprehensive response to the user's query. The LLM leveraged the context provided in the prompt and its own legal expertise to create a detailed and insightful answer.

3.3.5 Naive RAG

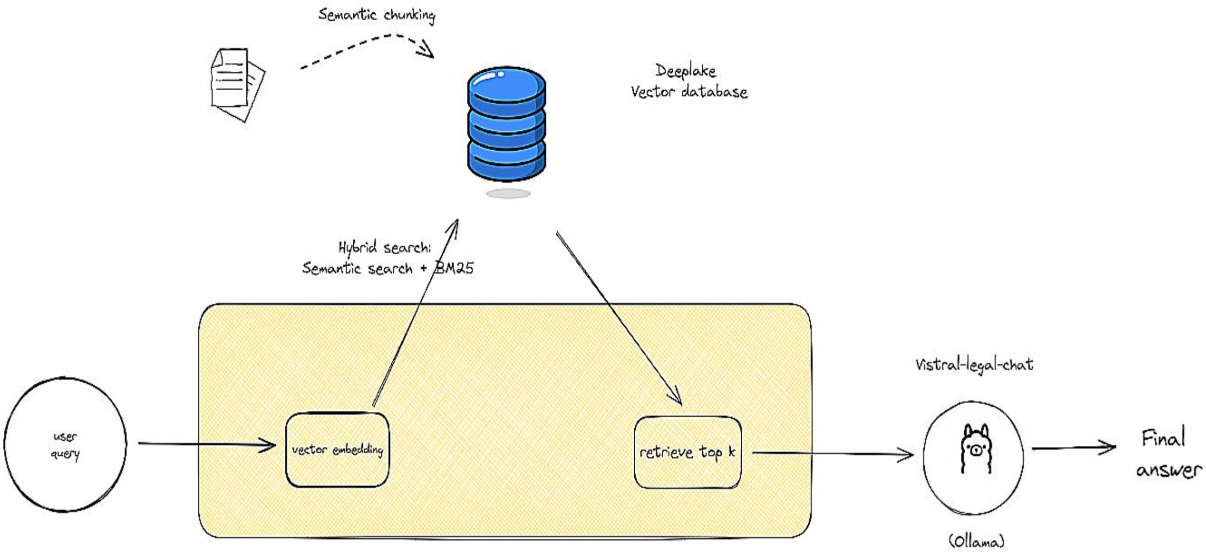

Figure 3-3: Naive RAG pipeline

This diagram illustrates the workflow of a Naive Retrieval-Augmented Generation (RAG) system. The user query is processed through a vector similarity search, retrieving relevant documents from a vector database. The top-k documents are then used by the LLM (Large Language Model) to generate the final answer.

3.3.6 Advanced RAG

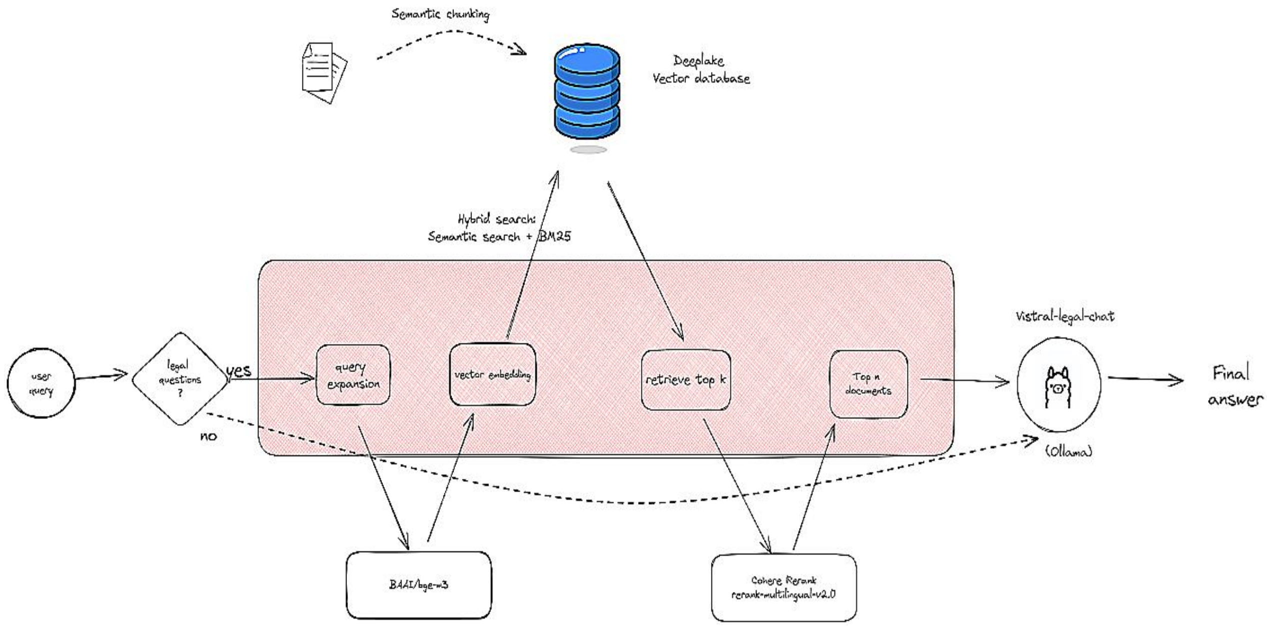

Figure 3-4: Advanced RAG pipeline

This diagram depicts the workflow of an Advanced Retrieval-Augmented Generation (RAG) system. The user query undergoes multiple stages, including query expansion and document retrieval using hybrid search methods. The retrieved documents are filtered, ranked, and refined before being used by the LLM to generate the final answer. This advanced approach incorporates additional steps such as context refinement to enhance the accuracy and relevance of the generated output.

3.3.7 Loading stage

3.3.7.1 Chunking strategy

A crucial step in the RAG pipeline is pre-retrieval step where the legal documents are segmented into smaller chunks to facilitate efficient retrieval. We explored several chunking strategies to optimize this stage:

- *Fixed-Size Chunking*: This strategy divides the document into chunks of a predetermined size, typically measured in tokens or characters. This approach is straightforward to implement but may lead to fragmented sentences or disjointed context within a chunk, potentially impacting retrieval quality.

- *Sentence Window Chunking*: This approach considers sentence boundaries while chunking. Each chunk consists of a predefined number of consecutive sentences. This strategy preserves sentence coherence, leading to better contextual understanding during retrieval.
- LLM-Assisted Chunking or semantic chunking: Advanced models like LLMs can be employed to analyze the document structure and identify meaningful segments for chunking. This allows for more dynamic and context-aware segmentation, potentially leading to improved retrieval accuracy.

For this specific legal question-answering task, we adopted the sentence window chunking strategy. This approach balances the need for manageable chunk sizes with the preservation of sentence-level context, which is crucial for understanding legal documents with their complex sentence structures and nuanced language.

3.2.1.2 Text embedding

The embedding phase plays a pivotal role in the RAG-Fusion process, transforming textual information into a format suitable for efficient retrieval. This involves generating vector representations, also known as embeddings, for both user queries and the legal documents within our knowledge base.

Figure 3-5: Text Embedding Process[2]

[2] https://www.pinecone.io/learn/vector-embeddings/

This embedding technique enables us to measure the similarity between a user's query and the legal documents. By comparing the embeddings of the query and each document, we can identify the most relevant documents based on their proximity in the vector space. The closer the embedding of a document is to the query's embedding, the more relevant it is considered to be.

The retrieved documents, along with the user's query and any additional context provided by the user, are then passed to a powerful language model (LLM). The LLM, equipped with this contextual information, generates a comprehensive and informative answer to the user's legal question. This process, known as RAG-Fusion, leverages the strengths of both retrieval and generation techniques to provide accurate and insightful legal information.

For the embedding phase, we selected the BAAI bge small v1.5. This model, developed by the Beijing Academy of Artificial Intelligence (BAAI), is a powerful and versatile embedding model specifically designed for natural language processing tasks.

The BAAI bge small v1.5 is a transformer-based embedding model trained on a massive dataset of text and code. Its architecture allows it to learn rich semantic representations of words and phrases, capturing subtle nuances in language and context.

We chose this model for the following reasons:

- Strong Performance: BAAI bge small v1.5 has consistently demonstrated impressive performance on various embedding tasks, including text similarity, semantic search, and question answering.
- Vietnamese Language Support: While not specifically trained on Vietnamese data, the model's ability to learn language nuances makes it adaptable to different languages, including Vietnamese.
- Resource Efficiency: The "small" variant of the model offers a balance between performance and computational resources, making it suitable for our application.

- *Open Source*: Its open-source nature facilitates easy integration and experimentation.

By leveraging the BAAI bge small v1.5 model for embedding, we aim to achieve efficient and accurate retrieval of relevant legal documents, paving the way for a more robust and effective Vietnamese legal question-answering system.

3.3.7.2 Deeplake Vectorstore

After generating embeddings for the legal, the next step involves storing these embeddings in a vector database for efficient retrieval during the RAG-Fusion process.

For this task, we opted for *a cloud-based vector database*. This is because cloud-based vector databases offer unparalleled scalability, effortlessly accommodating the growing size of our knowledge base as we incorporate more legal documents. Besides, cloud providers handle the complexities of managing and maintaining the vector database infrastructure, freeing us to focus on developing and refining our legal QA system. Not to mention that, cloud-based solutions often offer pay-as-you-go pricing models, making them cost-effective, especially for projects with fluctuating resource needs.

Cloud-based vector databases typically provide seamless integration with other cloud services and offer well-documented APIs for easy access and manipulation of the embedded data.

By leveraging a cloud-based vector database, we can ensure efficient storage and retrieval of the embedded legal documents, streamlining the RAG-Fusion process and enabling fast and accurate responses to legal queries.

In this project, we use ***Deep Lake*** Vector database by *Activeloop* as our vector database solution. DeepLake, a cutting-edge platform for managing and manipulating large datasets, offers several key advantages that align perfectly with our needs:

- High-Performance Vector Storage: DeepLake is specifically designed for storing and querying high-dimensional vectors, making it ideal for handling the embeddings generated from our legal documents. Its optimized storage and retrieval mechanisms ensure fast and efficient access to the embedded data.

- Integration with PyTorch and TensorFlow: DeepLake seamlessly integrates with popular deep learning frameworks like PyTorch and TensorFlow, simplifying the process of loading, transforming, and querying embedded data within our RAG-Fusion pipeline.

By leveraging DeepLake, we gain a powerful and efficient platform for managing the embedding of our legal documents, enabling us to build a robust and scalable Vietnamese legal question-answering system.

3.3.8 Index state

3.3.8.1 Query expansion

The query stage is crucial for bridging the gap between user intent and the retrieval of relevant legal documents. While users may provide a clear and concise query, many times their prompts are vague or lack specific keywords, making it difficult for the retriever to identify the most pertinent information within the knowledge base.

Example of a Vague Query:

Consider a user asking: "Tôi muốn biết về quyền lợi của người lao động trong trường hợp bị sa thải." (I want to know about the rights of workers in case of dismissal).

This query is relatively vague. While it mentions "rights of workers" and "dismissal," it lacks specific details about the type of dismissal, the worker's situation, or the relevant legal provisions. This lack of specificity makes it challenging for the retriever to pinpoint the most relevant documents within the legal knowledge base.

To address this challenge, we introduce a query transformation step before retrieval. This step aims to enhance the user's query by generating multiple, more specific queries that capture different facets of the user's intent.

Example of Query Transformation:

Using our previous example, the query transformation module could generate the following additional queries:

- "Quyền lợi của người lao động bị sa thải không lý do." (Rights of workers dismissed without cause).
- "Quyền lợi của người lao động bị sa thải do vi phạm hợp đồng." (Rights of workers dismissed due to contract violations).
- "Quyền lợi của người lao động bị sa thải do doanh nghiệp phá sản." (Rights of workers dismissed due to business bankruptcy).

By expanding the query set to include these more specific variations, we increase the likelihood of retrieving relevant documents that cover a wider range of scenarios related to the user's initial query.

This query transformation technique significantly improves the retrieval process by:

- Addressing Query Vagueness: It helps to clarify the user's intent by exploring various interpretations of the original prompt.
- Expanding Retrieval Scope: It increases the chances of retrieving relevant documents by searching for information related to a broader set of keywords and concepts.

By combining the original query with these transformed queries, we enhance the retrieval process, enabling our system to identify more relevant legal documents and

3.3.8.2 Hybrid search: vector search + BM25 search

To achieve robust and accurate retrieval of relevant legal documents for our Vietnamese legal chatbot, we implemented a hybrid search approach that combines the strengths of both vector search and BM25 search.

- ***Vector Search:*** This method leverages semantic meaning to find similar documents, going beyond simple keyword matching. It uses embedding models to represent legal documents and user queries as vectors, then calculates their similarity for retrieval. While efficient for semantic understanding, it can be computationally expensive and struggles with complex queries.

- ***BM25 Search***: A traditional keyword-based search algorithm, BM25 ranks documents based on keyword frequency and rarity within the dataset. It is fast, efficient, and easy to implement, but lacks semantic understanding and relies solely on keyword matching.

To address the limitations of each method, we implemented a hybrid approach which is the combination of the above two methods:

- *Vector Search* ensures semantic relevance, capturing the nuanced meaning of user queries.
- *BM25 Search* provides a fast and efficient way to rank documents based on keyword matching, ensuring relevance for queries with specific legal terms.

Combining these two methods allow us handle a wide range of legal queries, from those with clear keywords to those with more nuanced semantic meanings. Besides, it also provides accurate and relevant answers, ensuring that users find the most pertinent legal information for their needs.

3.3.9 Post retrieval

3.3.9.1 Metadata replacement

Metadata replacement is a crucial step in the post-retrieval step of our RAG-based Vietnamese legal chatbot. It enhances the retrieval process by incorporating contextual information from retrieval legal documents, improving the accuracy and relevance of the final answer generated by the chatbot.

During the retrieval stage, the retriever identifies n relevant legal documents based on the user's query. Each retrieved node includes valuable metadata, such as:

- Document title: inform us about the kind of law will be used
- Page number: help pinpoint specific legal provision within the document

- Date of publication: indicate the document’s relevance and potential changes overtime
- Window sentences: 5 sentences before and 5 sentences after the current content. This help to get more precise context for the chatbot.

The retrieved documents are replaced with the metadata. For example:

User Query: "Tôi muốn biết quyền lợi của người lao động khi bị sa thải?" (What are the rights of workers in case of dismissal?)

Retrieved Document: "Bộ luật Lao động, Điều 115, quy định về việc chấm dứt hợp đồng lao động và quyền lợi của người lao động trong các trường hợp này. Người lao động có quyền được bồi thường khi bị chấm dứt hợp đồng lao động mà không có lý do chính đáng." (The Labor Code of Vietnam, Article 115, addresses termination of employment contracts and the rights of workers in such situations. ultimately provide more accurate and comprehensive answers to user queries.

3.3.9.2 Rerank with Cohere

While retrieval mechanisms effectively extract multiple chunks from large documents, there's a chance of selecting irrelevant candidates. Reranking addresses this by re-evaluating and re-ordering search results to present the most relevant options. This process refines the final context provided to the LLM, focusing its attention on more concentrated information, ultimately boosting overall efficiency.

For this thesis, we decided to employ Cohere Reranker. Cohere, a leading startup in the AI space, offers a suite of state-of-the-art models for various NLP tasks, including RAG. We leverage their Reranker tool to further refine the quality of our retrieved documents. The Cohere Reranker enhances the accuracy and relevance of search results by leveraging the capabilities of a large language model (LLM) to assess the semantic similarity between retrieved documents and the user's query.

The process involves grouping documents into batches, which are then evaluated by the LLM. The LLM assigns relevance scores to each batch, ranking them based on their proximity to the user's query. Finally, the most relevant documents from all batches are aggregated to form the final retrieval response. This method ensures that the most pertinent information is highlighted and presented as the focal point of the search results.

By implementing Cohere Reranking, we aim to enhance the accuracy and relevance of the retrieved documents, providing the LLM with a more focused and informative context for generating accurate and comprehensive answers to user queries.

3.3.10 Query state

The query phase is where the chatbot processes user input and generates responses. To optimize this process for efficiency and accessibility, we employed two key strategies:

3.3.10.1 Quantization for efficient inference

We quantized the fine-tuned models (Vistral, Vinallama, PhoGPT) to 4-bit precision using the GGUF format. This quantization technique reduces the model's size significantly, enabling efficient inference even on CPUs with limited resources.

The quantization process converts the model's weights from 32-bit floating point number to 4-bit integers. This reduces the model's memory's footprint by a factor of 8, making it more efficient to load and process.

Next, GGUF format is a light-weight format for storing and loading quantized models. It optimized for efficient loading and execution, and can be inferenced with even CPUs.

By quantizing the models to 4-bit precision and using the GGUF format, we achieved significant improvements in inference speed and resource efficiency. This allowed us to deploy the chatbot on devices with limited computational resources, making it more accessible to a wider range of users.

After that, Ollama comes into play. Ollama is an open-source, modular runtime for running large language models (LLMs). It provides a lightweight and flexible platform for deploying and executing LLMs on various devices, including CPUs.

We leveraged Ollama's capabilities to efficiently execute the quantized LLMs during the query phase. Ollama's lightweight architecture and efficient execution engine enabled us to run the models on CPUs without significant performance degradation.

3.3.10.2 Generation

The generation stage is the heart of the RAG pipeline, where the user's query is processed and transformed into a comprehensive and informative answer. This stage leverages the reasoning ability of LLMs to generate human-like and factual responses based on the retrieved documents.

When the retrieved documents are processed using metadata replacement to expand the context of relevant sentences and reranked by Cohere to prioritize the most relevant and important content, this enriched context, along with user's original query, is combined into a specific prompt for the fine-tuned LLMs. Then, it is fed into the LLMs. The LLMs will process, understand and generate comprehensive and informative response.

For example:

User Query: "Tôi muốn biết quyền lợi của người lao động khi bị sa thải?" (What are the rights of workers in case of dismissal?)

Retrieved Document: "Bộ luật Lao động, Điều 115, quy định về việc chấm dứt hợp đồng lao động và quyền lợi của người lao động trong các trường hợp này. Người lao động có quyền được bồi thường khi bị chấm dứt hợp đồng lao động mà không có lý do chính đáng." (The Labor Code of Vietnam, Article 115, addresses termination of employment contracts and the rights of workers in such situations. Workers have the right to compensation when their employment contract is terminated without a valid reason.)

Prompt Construction: The chatbot combines the user query with the relevant sentence from the retrieved document, along with relevant metadata, to construct a prompt for the LLM.

Prompt: "Bộ luật Lao động, Điều 115, quy định về việc chấm dứt hợp đồng lao động và quyền lợi của người lao động trong các trường hợp này. Người lao động có quyền được bồi thường khi bị chấm dứt hợp đồng lao động mà không có lý do chính đáng. Tôi muốn biết quyền lợi của người lao động khi bị sa thải?" (The Labor Code of Vietnam, Article 115, addresses termination of employment contracts and the rights of workers in such situations. Workers have the right to compensation when their employment contract is terminated without a valid reason. What are the rights of workers in case of dismissal?)

LLM Response: "Theo Điều 115 Bộ luật Lao động, người lao động có quyền được bồi thường khi bị chấm dứt hợp đồng lao động mà không có lý do chính đáng. Quyền lợi cụ thể sẽ phụ thuộc vào lý do chấm dứt hợp đồng và điều kiện cụ thể của từng trường hợp." (According to Article 115 of the Labor Code, workers have the right to compensation when their employment contract is terminated without a valid reason. The specific rights will depend on the reason for termination and the specific circumstances of each case.

CHAPTER 4 EXPERIMENT

4.1 Fine-tune LLMs

4.1.1 Prepare data for fine-tuning

To effectively train our selected large language models (LLMs) for Vietnamese legal question-answering, a prompt-based instruction fine-tuning methodology was employed. This approach aimed to guide the LLMs towards generating human-like and legally accurate responses, emulating the expertise of a professional lawyer.

A meticulously crafted prompt template was designed to guide the LLMs in a role-playing scenario, where the model assumes the role of a professional lawyer providing legal advice to a client. The prompt template was structured as follows:

```
Bạn là một chuyên viên tư vấn pháp luật tại Việt Nam với nhiều năm kinh nghiệm và kiến thức chuyên sâu. Nhiệm vụ của bạn là cung cấp câu trả lời và tư vấn pháp lý cho các câu hỏi của người dùng.

### Câu hỏi:
{}

### Trả lời:
{}
```

The question-and-answer pair will be inserted into the designated prompt, and then it will be fed into the LLMs for instruction fine-tuning step.

4.1.2 Fine-tune with QloRA

To efficiently fine-tune our chosen 7B parameter LLMs (Vistral and Vinallama) for legal question-answering, we leveraged the QLoRA (Quantized Low-Rank Adaptation) technique. QLoRA allows for the effective fine-tuning of large models on a single T4 GPU, significantly reducing memory requirements and training time. This approach was essential for maximizing resource efficiency and accelerating our experimental process.

To facilitate a meaningful comparison of the performance between Vistral and Vinallama, the same hyperparameter set was used for each model during fine-tuning. This ensured that the differences in performance observed could be attributed to the underlying model architecture rather than variations in the training configurations.

The following table explains each hyperparameter used in fine-tuning:

Table 4-1: Hyperparameters Used for Fine-Tuning Vistral and Vinallama Models

Hyperparameter	Meaning	Value
r	Rank of the low-rank decomposition	8
Target modules	A list of specific layers within the LLM that will be fine-tuned using QLoRA.	["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj",]
Lora_alpha	The scaling factor for the LoRA weights.	8
Lora_dropout	The dropout rate applied to the LoRA weights.	0
bias	Controls whether to fine-tune the bias terms in the LoRA layers	None
use_gradient_checkpointing	Enables gradient checkpointing,	True
Random_state	ensure reproducibility of the training process.	2024
Use_rsload	Enables rank stabilized LoRA	False
Loftq_config	An optional configuration for the LoftQ technique	None

After successfully implementing QLoRA for fine-tuning, both Vistral and Vinallama were trained for one epoch with a maximum of 60 steps. The training process yielded significant improvements in the models' ability to handle Vietnamese legal question-answering.

The following table shows some information about the process of instruction fine-tuning:

Table 4-2: Instruction fine-tuning information

	Vistral	Vinallama	PhoGPT
No. of parameters (qLoRA)	20,971,520	12,517,376	4,423,680
Memory usage	5.5GB	5.5GB	3.9GB
Fine-tuning time	15:29	6:17	~10mins
Training_loss (step 60th)	**0.893900**	1.636900	1.63600

All models were converted into GGUF format for deployment and pushed to our hugging account.

4.1.3 Fine-tuned LLMs results

To assess the effectiveness of the fine-tuning process on Vistral and Vinallama, a comprehensive evaluation was conducted using a set of 50 carefully curated legal questions. These questions were specifically designed to test the models' ability to understand and respond to complex legal scenarios.

Evaluation metrics include:

BLEU (Bilingual Evaluation Understudy) Score: This metric measures the fluency and accuracy of generated text by comparing it to a set of reference answers. A higher BLEU score indicates a closer resemblance to the reference text, suggesting better fluency and faithfulness to the original legal information.

BertScore Evaluation: BERTScore utilizes pre-trained contextual embeddings from BERT to compare words between candidate and reference sentences using cosine similarity. Research has demonstrated that it aligns well with human assessments in both sentence-level and system-level evaluations.

Additionally, BERTScore calculates precision, recall, and F1 scores, making it valuable for assessing various language generation tasks.The table below show the results:

Table 4-3: Evaluation result of three models

Model	Bleu score	Precision	Recall	F1
PhoGPT 4b	20.802542062392376	0.6747	0.7263	0.7263
Vinallama 7b	21.606192202097755	0.6750	0.7313	0.6982
Vistral 7B	**31.817937925667252**	**0.6996**	**0.7604**	**0.7272**

Based on the results above, we see that Vistral 7b outperform the others in both Bleu score and LLM-based evaluations. In the next steps, we will further employ Vistral to build RAG system.

4.2 Experiment result

4.2.1 Manual evaluation

Manual evaluation of the RAG (Retrieval-Augmented Generation) model involves assessing the generated answers based on three key metrics: Grammar, Context Appropriateness, and Closeness to Ground Truth. These metrics provide a qualitative measure of the model's performance, allowing us to understand its strengths and weaknesses in generating coherent and accurate responses.

Grammar:

This metric evaluates the grammatical correctness of the generated answers. It checks for proper sentence structure, punctuation, verb tense consistency, and overall fluency of the text. A high score in this metric indicates that the generated answer is free of errors.

Context Appropriateness:

Context appropriateness measures how well the generated answer aligns with the context provided. This involves assessing whether the answer is relevant to the question posed and if it correctly incorporates information from the retrieved context. A high score for this statistic indicates that the model is producing accurate and contextually appropriate answers by efficiently utilizing the information that has been provided.

Closeness to Ground Truth:

This metric evaluates how closely the generated response and the ground-truth answer resemble each other. In order to make sure that the model's output is factually accurate and has a similar meaning. A higher score for this statistic means that, in terms of accuracy and content, the generated answer closely resembles the desired answer.

4.2.2 BERTScore

BERTScore is a assessment metric for natural language processing (NLP) tasks that uses contextual embeddings from pre-trained language models such as BERT (Bidirectional Encoder Representations from Transformers) to quantify the similarity between a candidate text and a reference text. In contrast to conventional measures like BLEU or ROUGE, which depend on surface-level matching (such n-gram overlap), BERTScore leverages the deep semantic understanding of language models to provide more nuanced and context-aware evaluations.

BERTScore calculates similarity by comparing the embeddings of tokens from the candidate and reference texts. The process involves the following steps:

Tokenisation and Embedding: After tokenizing the candidate and reference texts, a pre-trained BERT model processes their tokens to provide contextual embeddings for each token.

Token Alignment: Based on the cosine similarity between their embeddings, BERTScore identifies the token in the reference text that is most similar to each token in the candidate text. This process is performed in both directions (candidate-to-reference and reference-to-candidate).

Precision, Recall, and F1 Calculation:

Precision: The average similarity of each token in the candidate with its best match in the reference.

Recall: The average similarity of each token in the reference with its best match in the candidate.

F1-Score: The harmonic mean of Precision and Recall, providing a balanced measure of the similarity between the candidate and reference texts.

4.2.3 BLEU Score

The BLEU (Bilingual Evaluation Understudy) score is a widely used metric for assessing the quality of machine-generated text, particularly in the context of machine translation. BLEU was first presented by Kishore Papineni and his colleagues in 2002. It compares the n-grams, or word sequences, of the candidate text with those of human-generated reference translations. A score of one denotes a perfect match between the generated text and the reference. The value ranges from 0 to 1 depending how perfectly matched the generated text and the reference are.

Although BLEU has gained popularity as a benchmark for benchmarking machine translation systems, it is not without its drawbacks. Since the measure is sensitive to exact word matches, translations that are semantically correct but employ a different phrase may be penalized. Furthermore, BLEU only looks for surface-level matches and ignores the text's overall meaning and fluency. Notwithstanding these shortcomings, BLEU is a widely used method for comparing machine translation results because to its simplicity and computational easiness.

4.2.4 Ragas for RAG evaluation

Ragas is a framework helps evaluate Retrieval Augmented Generation (RAG) pipeline. RAG denotes a class of LLM applications that use external data to augment the LLM’s context. There are existing tools and frameworks that help you build these pipelines but evaluating it and quantifying your pipeline performance can be hard.

There are 4 metrics of RAGAS:

Faithfulness

This metric accesses the factual accuracy of the generated answer using the provided context. The generated answer is considered faithful if all its claims can be inferred from the provided context.

It is calculated from **generated answer** and **retrieved context** using a **critic llm.**

$$\text{Faithfulness score} = \frac{|\text{Number of claims in the generated answer that can be inferred from given context}|}{|\text{Total number of claims in the generated answer}|}$$

Figure 4-1: The formula of Ragas's faithfulness metric.[3]

Correctness

Answer correctness considers two key aspects: semantic similarity between the generated answer and ground-truth, as well as the factual similarity. This metric is calculated by combining factual correctness and semantic similarity between the generated answer and the ground truth.

Factual correctness is determined by identifying:

- True Positives (TP): Facts that are included in both the generated answer and the ground truth.
- False Positives (FP): Facts that are present in the answer generated but absent from the actual data.
- False Negatives (FN): Information that is included in the ground truth but absent from the produced response.

Based on these components, an F1 score is utilized to measure the factual correctness. Semantic similarity gauges how closely the generated response's meaning adheres to the original meaning.

[3] https://docs.ragas.io/en/latest/concepts/metrics/faithfulness.html

$$\mathbf{F1\ Score} = \frac{|\mathbf{TP}}{(|\mathbf{TP}| + 0.5 \times (|\mathbf{FP}| + |\mathbf{FN}|))}$$

Figure 4-2: The formula of F1 Score[4]

The final answer correctness score is a weighted average of the factual and semantic similarities, with adjustable weights for each component.

Context recall

Context recall calculates how much the retrieved context aligns with the ground-truth answer

To estimate context recall, each sentence in the ground truth answer is analyzed to see if it can be found in the retrieved context.

$$\text{context recall} = \frac{|\text{GT claims that can be attributed to context}|}{|\text{Number of claims in GT}|}$$

Figure 4-3: The formula of Ragas context recall metric[5]

Context precision

Context precision measures if all relevant items in the context are ranked high or not. It checks how precise is the context fetched.

Given the question, ground-truth answer, and retrieved context verify if the context was useful in arriving at the given answer, it done by prompting a critic llm.

$$\text{Context Precision@K} = \frac{\sum_{k=1}^{K} (\text{Precision@k} \times v_k)}{\text{Total number of relevant items in the top } K \text{ results}}$$

$$\text{Precision@k} = \frac{\text{true positives@k}}{(\text{true positives@k} + \text{false positives@k})}$$

Figure 4-4: The formula of Ragas context precision metric.[6]

4.2.5 Evaluation result

[4] https://docs.ragas.io/en/latest/concepts/metrics/answer_correctness.html
[5] https://docs.ragas.io/en/latest/concepts/metrics/context_recall.html
[6] https://docs.ragas.io/en/latest/concepts/metrics/context_precision.html

Table 4-4: Evaluation result using BERTScore and BLEU metrics with 100 samples

	Precision	Recall	F1	BLEU
Naive RAG	0.7339	0.8021	0.7649	0.4698
Fusion RAG	0.7554	0.8100	0.7758	0.5033
Phogpt	0.6747	0.7263	0.7263	0.2690
Vinallama	0.6705	0.7313	0.6982	0.3286
Vistral-legal-chat	0.6996	0.7604	0.7272	0.3831

After evaluating all five models, we can see Fusion RAG consistently performs better than the others across the board in all important measures. It produces 1.4% higher F1 score, 1% higher Recall, and 6.7% greater Precision than Naive RAG. Most notably, Fusion RAG's BLEU score is 7.1% higher than that of Naive RAG, indicating a significant improvement in generating text that closely matches the reference translations. In contrast, Phogpt, Vinallama, and Vistral-legal-chat lag behind, with Phogpt showing the lowest BLEU score, 42.6% lower than Fusion RAG, highlighting its lesser ability to match reference translations accurately.

Table 4-5: Evaluation result using Ragas with 30 samples

	Context precision	Context Precision	Faithfulness	Correctness
Naive RAG	0.5494	0.8389	0.7177	0.84385
Fusion RAG	0.7555	0.8722	0.8624	0.85529

With 30 samples, we see that Fusion RAG outperforms Naive RAG by 37.5% in Context Precision**,** 4% in Context Recall**,** and 20.2% in Faithfulness. Additionally, Fusion RAG has a 1.4% higher Correctness score than Naive RAG. These findings suggest that Fusion RAG is a better option in this situation because it produces outputs that are more faithful, accurate, and contextually appropriate.

Table 4-6: Evaluation using user-generated metrics with 100 samples

	Grammar	Contextual appropriate	Closeness to ground truths
Naive RAG	0.9561	0.2959	0.5164
Fusion RAG	0.9957	0.5755	0.7237

With 100 examples, we find that Fusion RAG outperforms Naive RAG once more, with a 4.1% higher score in Grammar. It also does 40.2% better in Closeness to Ground Truths and 94% better in Contextual Appropriateness. These significant improvements highlight Fusion RAG's capability to produce text that is grammatically correct, contextually appropriate, and closely aligned with the expected ground truth.

Overall, we see that Fusion RAG consistently demonstrates superior performance across all evaluations, making it the best-performing model among those tested.

4.2.6 Inference

4.2.6.1 User Interface Technology

We built the user interface for our chatbot using Django, a powerful and versatile web framework. Django allows us to seamlessly integrate backend logic with frontend presentation, making it an ideal choice for our project. The interface is composed of three main pages: the homepage, the chatbot page, and the contact us page.

Our homepage serves as the entry point to the application, providing users with an overview of the chatbot's functionality and its purpose. Our system's main component is the chatbot page, where users may communicate with the question-answering bot by entering their legal questions and getting answers produced by the RAG model. Lastly, the contact us page provides a means for users to communicate with our development team.

For the frontend, we employed HTML, CSS, and JavaScript on the front end to produce a dynamic and adaptable user interface. The site's interactive components are handled by JavaScript, which facilitates easy communication between the user interface and the backend logic. The sites' structural basis is provided by HTML, and the user interface is styled using CSS to make it both aesthetically pleasing and easy to use. Combining these technologies creates a user interface that is effective and efficient, making it easy to interact with the question-answering bot.

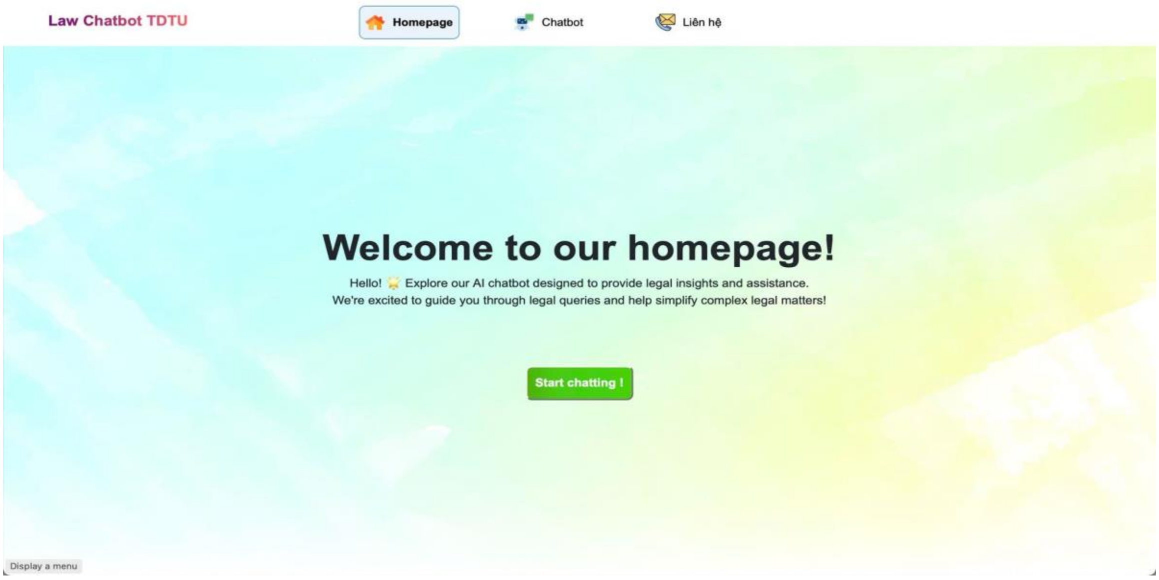

Figure 4-5: The interface of our Django-built chatbot. This is the homepage.

4.2.6.2 System Execution and Setup

To use our RAG-based question-answering bot, users need to follow a few steps to set up and run the system properly. First, it is crucial to establish a secure connection to the system, which we achieve using ngrok. Ngrok creates a secure tunnel to our local development environment, allowing external access to the Django application since our codebase is hosted on Lightning AI, which doesn’t allow for direct access to Django.

Here are the steps to set up and execute the system:

To set up and execute the system, first, launch Ngrok and connect it to the local environment, which generates a secure tunnel URL for remote access. Next, update the Django settings to include the Ngrok URL in the ALLOWED_HOSTS and CSRF_TRUSTED_ORIGINS, ensuring the application recognizes it as a legitimate host. After configuring the settings, start the Django server. Instead of accessing it via the local address, use the Ngrok-generated URL to interact with the application securely. This will allow one to access and interac with chatbot freely.

.

4.2.6.3 Sample Outputs

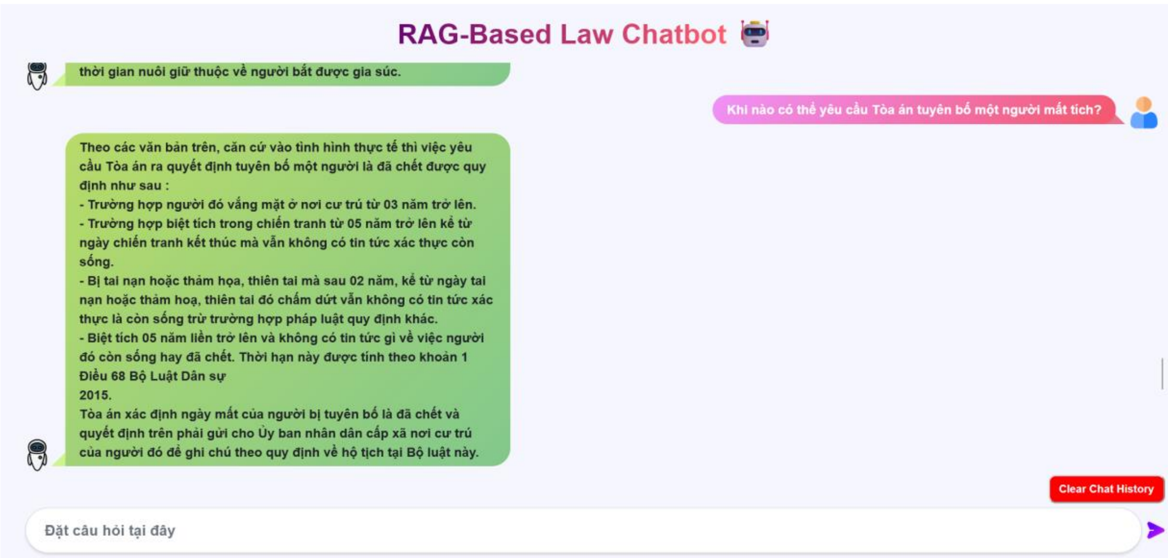

Figure 4-6: Sample output 1

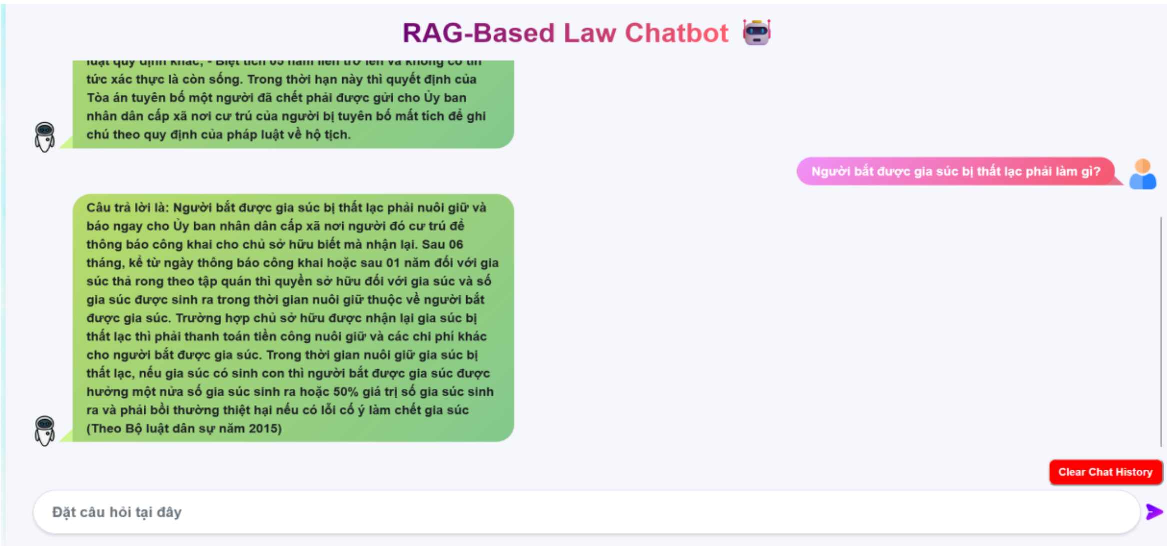

Figure 4-7: Sample output 2

Sample answers:

Table 4-7: Sample answers demonstrating the difference in capabilities between Naive RAG and advanced RAG

Question	Naive RAG	Advanced RAG
Khi nào có thể yêu cầu Tòa án tuyên bố một người mất tích?	Theo quy định tại Điều 63 Bộ luật dân sự năm 2015 thì khi một người biệt tích 02 năm liền trở lên, mặc dù đã áp dụng đầy đủ các biện pháp thông báo, tìm kiếm theo quy định của pháp luật về tố tụng dân sự nhưng vẫn không có tin tức xác thực về việc người đó còn sống hay đã chết thì theo yêu cầu của người có quyền, lợi ích liên quan, Tòa án có thể tuyên bố người đó mất tích.	Theo Điều 68 Bộ Luật Dân sự, khi một người biệt tích 02 năm liền trở lên, mặc dù đã áp dụng đầy đủ các biện pháp thông báo, tìm kiếm theo quy định của pháp luật về tố tụng dân sự nhưng vẫn không có tin tức xác thực nào về việc người đó còn sống hay đã chết thì Tòa án có thể tuyên bố người đó mất tích. Thời hạn 02 năm được tính từ ngày biết được tin tức cuối cùng về người đó; nếu không xác định được ngày có tin tức cuối cùng thì thời hạn này sẽ được tính từ ngày đầu tiên của tháng tiếp theo tháng có tin tức cuối cùng hoặc từ ngày đầu tiên của năm tiếp theo năm có tin tức cuối cùng.

Chiếm hữu là gì? Có mấy loại chiếm hữu	Chiếm hữu bao gồm chiếm hữu của chủ sở hữu và chiếm hữu của người không phải là chủ sở hữu. Việc chiếm hữu của người không phải là chủ sở hữu không thể là căn cứ xác lập quyền sở hữu, trừ trường hợp quy định tại các điều 228, 229, 230, 231, 232, 233 và 236 của Bộ luật này. Chiếm hữu ngay tình là việc c chiếm hữu mà người chiếm hữu có căn cứ để tin rằng mình có quyền đối với tài sản đang chiếm hữu; còn chiếm hữu không ngay tình là việc chiếm hữu mà người chiếm hữu biết hoặc phải biết rằng mình không có quyền đối với tài sản đang chiếm hữu.	Chiếm hữu được quy định tại Điều 176 Bộ luật dân sự năm 2015, theo đó thì việc chiếm hữu là việc chủ thể nắm giữ, chi phối tài sản một cách trực tiếp hoặc gián tiếp như chủ thể có quyền đối với tài sản. Việc chiếm hữu bao gồm hai loại đó là: - Chiếm hữu của người không phải là chủ sở hữu được quy định tại Điều 180 Bộ luật dân sự năm 2015, theo đó thì việc chiếm hữu của người không phải là chủ sở hữu là việc chiếm hữu mà người chiếm hữu biết hoặc phải biết rằng mình không có quyền đối với tài sản đang chiếm hữu. Việc chiếm hữu này không thể là căn cứ xác lập quyền sở hữu, trừ trường hợp quy định tại các Điều 228, 229, 230, 231, 232, 233 và 236 của Bộ luật dân sự năm 2015.

		- Chiếm hữu ngay tình được quy định tại Điều 180 Bộ luật dân sự năm 2015, theo đó thì việc chiếm hữu ngay tình là việc chiếm hữu mà người chiếm hữu có căn cứ để tin rằng mình có quyền đối với tài sản đang chiếm
Nghĩa vụ của chủ sở hữu trong việc thoát nước thải được quy định như thế nào?	Chủ sở hữu nhà, công trình xây dựng khác phải làm cống ngầm hoặc rãnh thoát nước để đưa nước thải ra nơi quy định, sao cho nước thải không chảy tràn sang bất động sản của chủ sở hữu bất động sản liền kề, ra đường công cộng hoặc nơi sinh hoạt công cộng.	Theo Điều 251 Bộ luật dân sự 2015, nghĩa vụ của chủ sở hữu nhà, công trình xây dựng khác phải làm cống ngầm hoặc rãnh thoát nước để đưa nước thải ra nơi quy định, sao cho không chảy tràn sang bất động sản của chủ sở hữu bất động sản liền kề, ra đường công cộng hoặc nơi sinh hoạt công cộng.
"Nếu chủ nợ không có bảo đảm, phải dùng tài sản nào để thanh toán khoản nợ?"	Chủ nợ không có bảo đảm là cá nhân, cơ quan, tổ chức mà trong trường hợp doanh nghiệp, hợp tác xã lâm vào tình trạng phá sản thì sẽ được yêu cầu người quản lý tài sản của doanh nghiệp, hợp tác xã phải thực hiện nghĩa vụ thanh toán khoản nợ.	Theo thông tin được cung cấp tại Điều 64 Luật Phá Sản năm 2015 thì khi một doanh nghiệp hoặc hợp tác xã mất khả năng thanh toán nghĩa vụ của mình với các chủ nợ, tài sản dùng để thanh toán cho những người này sẽ bao gồm ba nhóm sau đây : (i)Tài sản và quyền tài sản mà doanh nghiệp có tại thời điểm Tòa án nhân dân quyết định mở thủ tục phá sản; (ii) Tài sản và

		quyền tài sản có được sau ngày Toà án nhân dân ra Quyết định mở thủ tục phá sản; (iii) Giá trị của tài sản bảo đảm vượt quá khoản nợ có bảo đảm mà Doanh nghiệp, hợp tác xã phải thanh toán cho chủ nợ có bảo đảm.

CHAPTER 5 CONCLUSION

5.1 Result

This thesis has made significant contributions to the field of Vietnamese legal chatbots, demonstrating the potential of RAG-based approaches for providing accessible and accurate legal information to Vietnamese users. The key achievements of this research include:

Instruction Fine-tuning with qLoRA: This research successfully implemented qLoRA, a state-of-the-art technique for instruction fine-tuning of Large Language Models (LLMs). This fine-tuning process significantly enhanced the performance of the LLMs used in the chatbot, enabling them to better understand and respond to legal queries in Vietnamese.

Implementation of RAG for Vietnamese Legal Chatbot: This thesis has successfully implemented a Retrieval Augmented Generation (RAG) system specifically tailored for the task of Vietnamese legal question answering. This system leverages the strengths of both retrieval and generation techniques to provide comprehensive and accurate legal information to users.

Hands-on Experience with Advanced RAG Pipeline: The research involved hands-on experience with advanced RAG pipeline elements

Development of a User-Friendly Interface: A user-friendly interface was developed for the RAG-based Vietnamese legal chatbot, enabling users to easily interact with the system and access legal information in a convenient and accessible manner.

These achievements demonstrate the potential of RAG-based approaches for building effective Vietnamese legal chatbots. The research also provides valuable insights into the challenges and opportunities of utilizing LLMs and advanced RAG techniques for legal information retrieval in Vietnamese.

5.2 Significance

5.2.1 Scientific significance

This research holds significant scientific value for its contributions to Vietnamese legal technology and the broader field of natural language processing (NLP). The findings and outcomes of this work have implications for several areas:

- Expanding Legal Question-Answering Data Applications: Building a Vietnamese legal chatbot using RAG required creating a comprehensive legal question-answering dataset. This dataset has wider applications beyond chatbot development, including legal text summarization, legal document classification, legal information extraction, and even legal translation.
- Fine-tuned LLMs for Legal Applications: We successfully fine-tuned LLMs (Vistral, Vinallama, PhoGPT) on Vietnamese legal data. These fine-tuned models, publicly available on Hugging Face, offer a valuable resource for the NLP community. Researchers and developers can now use these models to build new legal applications or improve existing ones.
- Advancement of Vietnamese Legal Chatbots: This research paves the way for more sophisticated and user-friendly Vietnamese legal chatbots. The successful implementation of RAG for legal question answering in Vietnamese shows the potential for:
    - Increased access to legal information: Making legal information and guidance more accessible to Vietnamese users.
    - Improved legal literacy: Empowering Vietnamese citizens with a better understanding of their legal rights and obligations.
    - Enhanced legal research: Streamlining legal research and providing efficient access to relevant legal resources.

By contributing to these areas, our research fosters a more accessible and equitable legal landscape in Vietnam. This empowers individuals and businesses with the knowledge and tools needed to navigate the legal system effectively.

5.2.2 Practical significance

This research holds significant practical significance by addressing a critical need for accessible and reliable legal information in Vietnam. The development of a Vietnamese legal chatbot offers several practical benefits:

- Real-Time Legal Consultation: The chatbot provides real-time question-answering capabilities, allowing Vietnamese users to receive immediate legal information and guidance. This timely access to legal knowledge empowers individuals to make informed decisions and navigate legal situations confidently.
- Improved Legal Literacy: By providing easily accessible legal information, the chatbot contributes to increased legal literacy among Vietnamese citizens. This empowers individuals to understand their rights and obligations, enabling them to proactively protect themselves and engage more effectively with the legal system.
- Leveraging Legal Knowledge: The chatbot allows users to leverage the vast body of Vietnamese legal knowledge. It provides a convenient and user-friendly platform to access legal information that might otherwise be difficult to find or understand. This access to legal knowledge helps users make more informed decisions and navigate legal situations with greater confidence.
- Cost-Effective Legal Information: The chatbot offers a cost-effective alternative to traditional legal consultations. By providing free or low-cost access to legal information, it empowers individuals who may not have the resources to afford professional legal advice.

The practical significance of this research lies in its ability to bridge the gap between legal knowledge and everyday life in Vietnam. The chatbot provides a valuable tool for empowering Vietnamese citizens with the legal information they need to make informed decisions and advocate for their rights.

5.3 Difficulties

While this research has yielded significant results, it faced several challenges and limitations during its development. These challenges highlight the complexities of building a robust and effective Vietnamese legal chatbot using RAG:

- **Steep Learning Curve with LLaMaIndex**: Getting started with LLaMaIndex, a powerful tool for managing and indexing large datasets, involved a significant learning curve. This required substantial time investment to understand the framework, its various components, and how to effectively integrate it into the RAG pipeline.
- **Limited Resources for Fine-Tuning and Inference**: Fine-tuning LLMs for legal tasks and deploying the chatbot for inference requires significant computational resources, including powerful GPUs and large amounts of data. These resources were limited, posing challenges in achieving optimal model performance and scaling the chatbot for real-world applications.
- **Resource-Intensive RAG and Latency**: Implementing advanced RAG techniques, such as vector search and metadata replacement, requires substantial computational resources and processing time. This can lead to increased latency during inference, impacting the user experience and the real-time responsiveness of the chatbot.
- **Bias in Evaluation**: Evaluating the performance of the RAG pipeline presented challenges due to the inherent biases present in the legal data and the language model itself. Ensuring a fair and unbiased evaluation of the chatbot's performance requires careful consideration of potential biases and their impact on the system's accuracy and fairness.

5.4 Future work

Future work should focus on addressing these limitations by:

Developing more efficient and resource-friendly RAG techniques and exploring alternative architectures and algorithms that reduce computational demands without compromising performance.

Improving access to computational resources: Seeking partnerships with organizations or institutions that can provide the necessary resources for fine-tuning and deploying large language models for legal tasks.

Addressing bias in legal data and language models: Developing robust methods for identifying and mitigating bias in legal datasets and the language models used for legal question-answering.

By overcoming these challenges, we can further advance the development of Vietnamese legal chatbots, creating more accessible and equitable access to legal information.

REFERENCES

Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2023). *Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection*. Retrieved from https://arxiv.org/abs/2310.11511

Chang, u., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., . . . Xie, X. (2024). *A Survey on Evaluation of Large Language Models*. Retrieved from https://dl.acm.org/doi/10.1145/3641289

Chen, J., Xiao, S., Zhang, P., Luo, K., Lian, D., & Liu, Z. (2024). *BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation*. Retrieved from https://arxiv.org/abs/2402.03216

Ding, N., Qin, Y., Yang, G., Wei, F., Yang, Z., Su, Y., . . . Zheng, H.-T. (2022). *Parameter-efficient fine-tuning of large-scale pre-trained language models*. Retrieved from https://www.nature.com/articles/s42256-023-00626-4

Doan, V. T., Truong, Q. T., Nguyen, D. V., Nguyen, V. T., & Luu, T. N. (2023). *Efficient Finetuning Large Language Models For Vietnamese Chatbot*. Retrieved from https://ieeexplore.ieee.org/abstract/document/10288647

Es, S., James, J., Espinosa-Anke, L., & Schockaert, S. (2023). *RAGAS: Automated Evaluation of Retrieval Augmented Generation*. Retrieved from https://arxiv.org/pdf/2309.15217

Gao, M., Hu, X., Ruan, J., Pu, X., & Wan, X. (2024). *LLM-based NLG Evaluation: Current Status and Challenges*. Retrieved from 2402.01383

Iusztin, P. (2024). *The 4 Advanced RAG Algorithms You Must Know to Implement*. Retrieved from https://decodingml.substack.com/p/the-4-advanced-rag-algorithms-you

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d., . . . T. (n.d.). *Mistral 7B*. Retrieved from arXiv:2310.06825

Krampah, P. (2024). *Agentic RAG with Llama-Index | Router Query Engine #01. Medium*. Retrieved from https://medium.com/aimonks/agentic-rag-with-llama-index-router-query-engine-01-381e83a418af

Kumawat, T. (2024, February). *Optimizing Language Model Fine-Tuning with PEFT, QLORA Integration, and Training Time Reduction using Unsloth: A Comprehensive Guide*. Retrieved from https://medium.com/@tejpal.abhyuday/optimizing-language-model-fine-tuning-with-peft-qlora-integration-and-training-time-reduction-04df39dca72b

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., . . . Kiela, D. (2020). *NIPS papers*. Retrieved from Retrieval-Augmented Generation for NLP tasks: https://proceedings.neurips.cc/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf

Li, H., Su, Y., Cai, D., Wang, Y., & Liu, L. (2022). *A Survey on Retrieval-Augmented Text Generation*. Retrieved from https://arxiv.org/pdf/2202.01110

Ma, Y., Cao, Y., Hong, Y., & Sun, A. (2023). *Large Language Model Is Not a Good Few-shot Information Extractor, but a Good Reranker for Hard Samples!* Retrieved from https://doi.org/10.18653/v1/2023.findings-emnlp.710

Nguyen, D. Q., Nguyen, L. T., Tran, C., Nguyen, D. N., Phung, D., & Bui, H. (2024). *PhoGPT: Generative Pre-training for Vietnamese*. Retrieved from https://arxiv.org/pdf/2402.01383

Nguyen, Q., Pham, H., & Dung, D. (2023). *VinaLLaMA: LLaMA-based Vietnamese Foundation Model*. Retrieved from arXiv: https://arxiv.org/pdf/2312.11011

Pagnoni, T. D., Holtzman, A., & Zettlemoyer, L. (2023). *QLoRA: Efficient Finetuning of Quantized LLMs*. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html

Rackauckas, Z. (2024). *RAG-FUSION: A NEW TAKE ON RETRIEVAL-AUGMENTED GENERATION*. Retrieved from https://aircconline.com/ijnlc/V13N1/13124ijnlc03.pdf

WANG, L., MA, C., FENG, X., ZHANG, Z., YANG, H., & ZHANG, J. (2024). *A survey on large language model based autonomous agents*. Retrieved from https://doi.org/10.1007/s11704-024-40231-1

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020). *BERTScore: Evaluating Text Generation with BERT*. Retrieved from https://arxiv.org/pdf/1904.09675