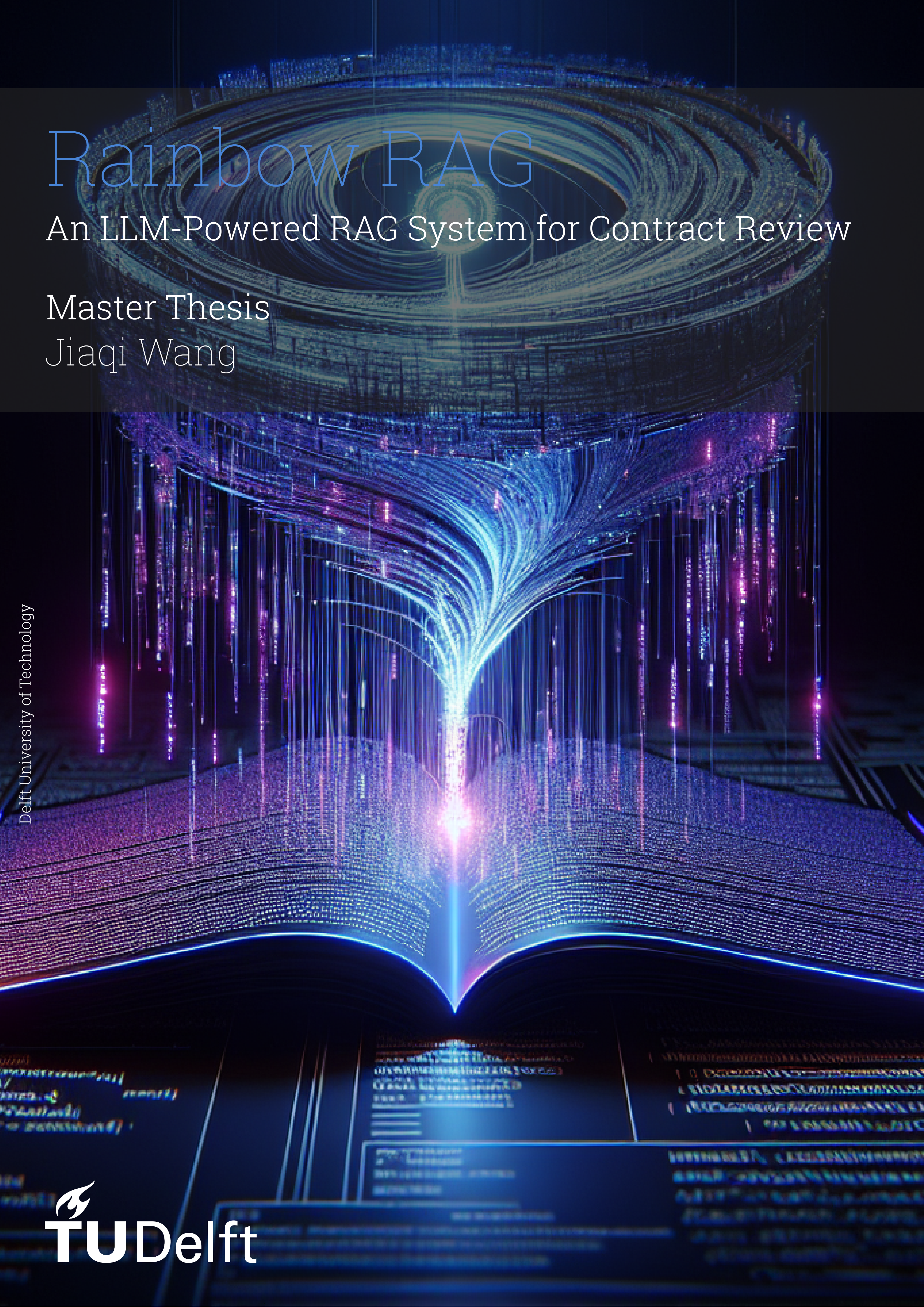


Rainbow RAG

An LLM-Powered RAG System for Contract Review

Master Thesis

Jiaqi Wang

Rainbow RAG

An LLM-Powered RAG System for Contract Review

by

Jiaqi Wang

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Wednesday November 20, 2024 at 13:45 PM.

Student number:	5679532	
Project duration:	December 1, 2023 – November 20, 2024	
Thesis committee:	Dr. R. Hai,	TU Delft, supervisor
	Dr. S. Pera,	TU Delft
	Dr. H. Wang,	TU Delft

Cover: Canadarm 2 Robotic Arm Grapples SpaceX Dragon by NASA under CC BY-NC 2.0 (Modified)
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

Preface

As a master's student in Computer Science at TU Delft, I present this thesis on leveraging Large Language Models in Contract Review, a collaborative research project between TU Delft and Sue B.V.. My journey in Natural Language Processing, which began during my bachelor's studies, coupled with the recent advances in large language models, naturally led me to this research domain. The opportunity to pursue this specific direction materialized through my connection with Sue BV.

This research introduces a novel approach to contract review using Large Language Models in a zero-shot setting. The core contribution lies in the development of an architecture that combines Knowledge Graphs with Retrieval Augmented Generation, resulting in enhanced performance. This work targets academic researchers and software developers with a computer science background, particularly those specialising in Natural Language Processing.

What began as an ambitious project evolved through various stages of literature review, experimental design, and iterative improvements, ultimately extending beyond the initial timeline to ensure comprehensive results. Through careful methodology and persistent refinement, I developed and validated an innovative approach to tackle the challenges in automated contract review.

I extend my sincere gratitude to my thesis supervisor, Dr. Rihan Hai, my company supervisor, Ir. Nathan Keyaerts, who provided both the research opportunity and experimental funding, and my daily supervisor, Dr. Ziyu Li (congratulations on your recent Ph.D.). Their consistent guidance and professional support were instrumental throughout this journey. I also appreciate Dr. Sole Pera and Dr. Huijuan Wang for their valuable participation in the thesis committee.

My peers in the Computer Science programme - Yufan Tang, Baitian Zhang, Haoran Wang, Karthik Prakash, Safouane El Hilali and Wenjie Zhao - deserve special mention for their continuous encouragement and support. Finally, I am deeply grateful to my family and girlfriend, whose unwavering love and support have been fundamental to my academic achievements.

Jiaqi Wang
Delft, November 2024

Abstract

Contract review is a critical yet time-consuming process in legal practice, with significant financial implications when errors occur. While Large Language Models (LLMs) have shown promise in legal document processing, they still face challenges with lengthy contracts and complex legal relationships. This research presents an advanced approach to automated contract review by integrating knowledge graphs into Retrieval-Augmented Generation (RAG) frameworks, addressing the limitations of current methodologies.

Through a comprehensive literature review of contract review automation and RAG systems, we conducted systematic experiments comparing RAG approaches with LLMs' in-context learning capabilities. Our empirical analysis validated that RAG-based methods significantly enhance long-context text analysis and information extraction in legal documents, particularly in terms of accuracy and consistency.

Building on these findings, we extensively investigated optimization techniques for the RAG retrieval phase, recognizing its critical role in contract review accuracy. Our experimental evaluation encompassed various chunking strategies, query expansion methods, and re-ranking approaches, establishing best practices for legal document processing.

Our primary contribution is a novel KG-RAG system that enhances contextual understanding in legal document analysis. We evaluate our approach using the Contract Understanding Atticus Dataset (CUAD) and ContractNLI dataset, demonstrating improved performance over traditional RAG implementations and long-context models. The research also explores optimal chunking strategies and investigates the efficiency-effectiveness trade-offs between different model architectures.

Results indicate that our KG-enhanced RAG framework achieves superior performance in identifying and analyzing complex legal relationships while maintaining computational efficiency. The integration of knowledge graphs particularly excels in capturing hierarchical and cross-referential relationships within legal documents, a crucial aspect often overlooked by conventional approaches.

This work advances the field of legal AI by providing a more robust and context-aware approach to contract review, while offering practical insights for implementing AI systems in legal practice. Our findings suggest promising directions for future research in legal document processing, particularly in areas requiring deep contextual understanding and relationship modeling.

Keywords: Legal AI, Contract Review, Knowledge Graphs, Retrieval-Augmented Generation, Large Language Models

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
NLP	Natural Language Processing
LLMs	Large Language Models
RAG	Retrieval Augmented Generation
KG	Knowledge Graph
LPOs	Legal Process Outsourcers

1 Introduction

Contracts form the cornerstone of partnerships and collaborations, with the inherent risks therein being a major concern for all parties involved [68]. Unidentified risks can lead to costly disputes, both in terms of time and finances. For instance, in the construction industry, the ARCADIS report [4] reveals that in 2021 alone, the average construction dispute resulted in losses of 52.6 million US dollars and lasted 15.4 months. These statistics underscore the critical importance of proper contract review for business success.

Hendrycks et al. [27] define contract review as the thorough examination of a contract to understand the rights and obligations it imposes, as well as its potential impact. This process spans from basic tasks like identifying and documenting relevant clauses to complex risk assessments.

At its core, contract review includes ”contract analysis,” where legal professionals meticulously search through extensive contracts to locate and interpret critical information. This involves identifying clauses, understanding their content, and noting their location within the document. Key details such as contract duration, termination dates, and specific clause types, like anti-assignment or most favored nation clauses, are extracted during this phase.

The more advanced aspect, termed ”counseling,” involves senior legal practitioners who leverage their deep industry knowledge to assess risks and offer strategic advice. This high-level analysis considers industry norms, business models, risk tolerance, and company priorities, going beyond literal clause interpretation to evaluate broader business implications.

Our research aims to use machine learning models to automate contract review and simpler aspects of contract analysis. This automation targets repetitive tasks like identifying and extracting relevant information, freeing legal professionals to focus on more complex, high-value work. We prioritize these tasks because they are more structured and easier to evaluate, given the scarcity of comprehensive datasets for complex analysis. This approach lays the groundwork for reliable, incremental improvements in legal automation.

Moreover, it is important to note that we aim to develop a system suitable for all clients while maintaining the privacy and security of their contracts. This requirement necessitates the ability to deploy such a system offline, and also means that fine-tuning models for each customer is not feasible, despite the potential for improved performance. These constraints further complicate the challenge of developing an effective, universally applicable contract review system. However, addressing these challenges is crucial to ensure widespread adoption and practical implementation of automated contract review tools across various industries and organizations.

1.1. Previous Attempts

Traditionally, this task has been accomplished primarily by human legal professionals, including junior lawyers, senior lawyers, or legal process outsourcers (LPOs). However, manual review is not only slower and more expensive [43] but also prone to errors [36].

Previous attempts to computerize contract review have primarily relied on two approaches: rule-based natural language processing (NLP) methods [33] and machine learning (ML) algorithms [13, 12]. However, these methods have shown significant limitations when confronted with the intricate language comprehension required for analyzing contract clauses. Rule-based NLP, while precise, suffers from inflexibility and the immense challenge of predefining rules to cover all possible scenarios [25]. ML algorithms, conversely, demand substantial high-quality data for training, necessitating considerable time investment in data collection, cleaning, and labeling. Furthermore, both approaches struggle to effectively capture semantic and contextual information, which severely constrains their performance in text understanding and processing [25, 70].

The advent of Large Language Models (LLMs) has revolutionized traditional NLP tasks, such as text classification, sentiment analysis, and machine translation. These models, boasting billions of parameters [50, 63, 7, 3, 30, 49], have outperformed specialized models designed for specific tasks and continue to evolve with increasing size and output quality. They can even support question answering with some reasoning capabilities [66].

Given their impressive capabilities, LLMs present a potentially feasible solution for the task of contract review. Their ability to process complex language and vast amounts of text suggests that they could assist in identifying relevant clauses, assessing risk factors, and streamlining the review process. This opens up the possibility of automating not just basic clause identification but also more nuanced analyses, thereby potentially transforming the scope and efficiency of contract review.

However, despite the expanding context windows of LLMs, which can now accommodate up to 128k [1][62] or even 200k [2] tokens—sufficient for most individual documents and their corresponding checklists—these models can still falter. They may overlook crucial information or produce inaccurate responses within the given context [53], particularly when dealing with lengthy or complex contracts. This limitation underscores the need for a more robust approach that can ensure comprehensive and accurate analysis of legal documents.

1.2. Research Questions

To address these shortcomings, our project proposes leveraging LLMs and Retrieval Augmented Generation (RAG) frameworks. This approach aims to overcome the limitations of previous methods by combining the advanced language understanding capabilities of LLMs with the ability to retrieve and integrate relevant external knowledge through RAG [38].

This paper explores the potential of enhanced RAG frameworks to revolutionize legal document handling and review processes, with a particular focus on improving the efficiency and accuracy of information retrieval and analysis. Our research investigates three key questions:

1. **How do RAG methodologies compare to long-context models in terms of effectiveness for legal document processing?**
2. **What is the impact of various enhancement techniques on the performance of RAG systems in contract review, and how do they affect both effectiveness and efficiency?**
3. **Can the integration of knowledge graphs, as an alternative to traditional vector databases, significantly improve RAG frameworks for legal information retrieval, given the structured nature of legal documents, such as contracts?**

By addressing these questions, we aim to contribute to the development of more robust and adaptable systems for legal document analysis, potentially transforming how legal professionals interact with and extract insights from vast corpora of legal texts.

1.3. Contributions

By addressing these research questions, our work aims to bridge the gap between current automated contract review capabilities and the comprehensive analysis traditionally performed by human legal experts. Our goal is to offer a more inclusive, efficient, and accurate tool for legal professionals and busi-

[1]https://platform.openai.com/docs/models/gpt-4o
[2]https://docs.anthropic.com/en/docs/about-claude/models

nesses, underscoring the transformative potential of technology in overcoming the challenges faced by the legal industry in contract review and analysis.

Our study makes several significant contributions to the field of legal document analysis, particularly in the domain of contract review:

1. **Enhanced RAG Pipeline for Contract Review:** We develop a novel pipeline specifically tailored for contract review, incorporating advanced chunking and retrieval techniques. To validate our approach, we conduct a comprehensive evaluation using the Contract Understanding Atticus Dataset (CUAD) [27], providing empirical evidence of our system's effectiveness.
2. **Comparative Analysis of RAG vs. Long-Context Models:** We present a detailed comparison between RAG-based approaches and long-context models across various model sizes. This analysis, conducted using the ContractNLI dataset, offers insights into the trade-offs between efficiency and effectiveness, considering both model size and the presence or absence of RAG components.
3. **Knowledge Graph Integration in RAG Frameworks:** We explore the potential of integrating knowledge graphs into the retrieval stage of RAG systems. This novel approach aims to provide richer contextual information, potentially enhancing the quality and relevance of generated outputs in legal document analysis.

These contributions collectively advance the state-of-the-art in legal AI, offering new methodologies for more accurate and efficient contract review processes, while also providing valuable insights into the comparative performance of different model architectures in this domain.

1.4. Thesis Outline

The rest of this report is structured as follows. Chapter 2 provides definitions and background knowledge relevant to the concepts employed in this thesis. Following this, in Chapter 3, related research will be dicussed, posing a research gap between current research and our problem. An formal formulation of the problem and our proposed solutions are explained in Chapter 4. In the next chapter we will discuss our further improvement which is knowledge-graph-based RAG system. Chapter 6 outlines the experimental setup and methodology used to address our research questions, including the design of our experiments, and then discuss the our results. Finally, in Chapter 8, we conclude with a discussion of the findings, implications, and potential directions for future work.

2 Background

2.1. Contract Review

If we look up in the Cambridge dictionary[1], the definition of contract is expressed as ”a legal document that states and explains a formal agreement between two different people or groups, or the agreement itself”. However, there is no serious definition of contract review yet.

Hendrycks et al. define it as a crucial process in both business and legal spheres [27], involving meticulously examining contractual documents to grasp the rights and obligations of the parties involved, to understand the potential impacts on individuals or companies entering into agreements.

Sometimes, extra work is also within the scope of contract review, such as ambiguous clause identification and modification[5], vaguely statements clarification[26]. Despite its significance, contract review is often viewed as one of the most repetitive and tedious tasks in the legal profession, typically assigned to junior associates. It’s a costly process that many consider an inefficient use of legal professionals’ skills, highlighting the need for more effective approaches.

2.2. Large Language Models (LLMs)

Large Language Models (LLMs) represent a significant advancement in natural language processing (NLP) and artificial intelligence (AI). These models are based on neural networks and trained on vast datasets, allowing them to understand and generate human-like text across diverse tasks and domains. They form the backbone of various AI applications, including contract review, due to their ability to analyze and process complex legal documents.

2.2.1. Architecture and Training

LLMs have evolved significantly, particularly with the introduction of the attention mechanism, which revolutionized their ability to handle complex linguistic structures. This mechanism, first proposed by Vaswani et al. [64], enables models to focus on different parts of the input when processing text, thereby capturing long-range dependencies more effectively.

Models such as BERT (Bidirectional Encoder Representations from Transformers) [17] utilize an encoder-based architecture that processes text bidirectionally, considering both preceding and succeeding context. This is achieved through masked language modeling, where the model predicts randomly masked words, fostering a deep contextual understanding of language.

Conversely, the GPT (Generative Pre-trained Transformer) series [55, 11] popularized decoder-only architectures, where the model generates text by predicting each token sequentially based on its preceding tokens. This structure aligns well with generation tasks, such as drafting and summarizing legal documents.

[1]https://dictionary.cambridge.org/dictionary/english/contract

Training LLMs involves two key stages: pre-training and fine-tuning. During pre-training, models learn general language patterns from large-scale, unlabeled text data through self-supervised learning methods like next-token prediction. Fine-tuning then adapts these models to specific tasks using smaller, labeled datasets.

One defining characteristic of LLMs is their scale. Recent models like GPT-3 [11] boast hundreds of billions of parameters, enabling superior performance in various NLP tasks, including few-shot and zero-shot learning scenarios [11].

2.2.2. Limitations

While LLMs have demonstrated substantial capabilities, they face several notable limitations that affect their deployment in specialized domains like law:

- **Hallucination**: LLMs may generate text that appears plausible but is factually inaccurate [45].
- **Bias**: They can perpetuate biases present in their training data, potentially impacting fairness in legal applications [9].
- **Lack of reasoning**: Although proficient in pattern recognition, LLMs often struggle with tasks requiring deep reasoning or understanding of legal principles [42].
- **Context window limitations**: Most LLMs are constrained by a fixed context window, limiting their ability to process lengthy documents typical in the legal domain [61].
- **Resource intensity**: Training and deploying large-scale LLMs require considerable computational resources, which may be prohibitive for some applications [51].
- **Temporal constraints**: The knowledge encoded in LLMs is static and cannot automatically update with new legal precedents or regulations [38].

2.2.3. Recent Developments

Recent advancements have focused on enhancing the capabilities of LLMs to address their inherent limitations. Instruction-tuned models like GPT-3.5 and GPT-4 [50] exhibit improved abilities to follow complex instructions and maintain coherent dialogues. Techniques such as retrieval-augmented generation (RAG) [38], which combines LLMs with external knowledge bases, and chain-of-thought prompting [66] are being developed to improve reasoning capabilities. Additionally, frameworks like constitutional AI [8] are being explored to align LLM outputs with human values more closely.

2.3. Retrieval Augmented Generation (RAG)

LLMs have demonstrated impressive capabilities across various tasks, with their knowledge acquired and stored within the model parameters during pre-training and fine-tuning processes [56, 52]. However, this approach has inherent limitations, as the factual knowledge embedded in the model parameters is static and may not reflect the most up-to-date information or capture domain-specific nuances accurately.

To address these drawbacks, the Retrieval Augmented Generation (RAG) framework was introduced by Lewis et al. [38]. RAG aims to bridge the knowledge gap in LLMs by incorporating external knowledge bases during the generation process, allowing models to access and leverage up-to-date and domain-specific information. This approach enhances the accuracy and relevancy of the generated outputs by providing access to the latest information from external sources.

The RAG framework operates by combining two key components: a retrieval mechanism and a generation model. When presented with a query or prompt, the retrieval component first searches a large corpus of documents or knowledge base to find relevant information. This retrieved information is then provided as additional context to the generation model, typically an LLM, which uses it to produce a response. This approach allows the model to dynamically access and incorporate external knowledge, effectively expanding its information base beyond what was learned during training. By doing so, RAG enables more accurate, up-to-date, and context-aware responses, particularly in scenarios requiring specific or current information that may not be present in the model's static knowledge.

2.4. Knowledge Graph

- Explanation of knowledge graphs and their structure
- Basic concepts: nodes, edges, triples

Knowledge graphs have emerged as a critical component in numerous enterprises and applications, revolutionizing the way information is structured and utilized. Fensel et al. [20] provide a comprehensive definition, describing knowledge graphs as ”very large semantic nets that integrate various and heterogeneous information sources to represent knowledge about certain domains of discourse.” This definition underscores the versatility and power of knowledge graphs in synthesizing diverse data sources to create a cohesive representation of knowledge.

The fundamental structure of a knowledge graph is composed of three key elements:

- Nodes: Representing entities or concepts
- Edges: Depicting relationships between nodes
- Triples: Consisting of subject-predicate-object statements that form the basic unit of information

This structure allows for the efficient representation and interconnection of complex information, enabling sophisticated querying and inference capabilities.

The development and proliferation of knowledge graphs have been significantly facilitated by advancements in the Semantic Web and linked data technologies. Several prominent open knowledge graphs have emerged as central datasets on the Semantic Web, providing a wealth of cross-domain factual knowledge [20]. These include:

- DBpedia: Extracts structured data from Wikipedia[6]
- Wikidata: A community-curated knowledge base[65]
- YAGO: Combines Wikipedia content with WordNet[59]

These open knowledge graphs serve as invaluable resources for researchers and developers, offering extensive collections of structured, interlinked data that can be leveraged for various applications.

The versatility and power of knowledge graphs in representing and interconnecting complex information have made them indispensable in modern data-driven applications. As research in this field continues to advance, knowledge graphs are expected to play an increasingly crucial role in driving intelligent systems and enabling more sophisticated data analysis and decision-making processes.

2.5. Artificial Intelligence in Legal Field

The application of Artificial Intelligence (AI), particularly Large Language Models (LLMs), in the legal domain has gained significant traction in recent years. Researchers are exploring various legal tasks and challenges, leveraging the sophisticated language understanding and generation capabilities of LLMs to assist legal professionals and enhance legal processes.

2.5.1. Fine-Tuned Language Models for the Legal Domain

Building on the success of general-purpose LLMs, there is a growing interest in fine-tuning these models for specific legal tasks. An exemplary case is SaulLM-7B, a pioneering 7 billion parameter large language model tailored explicitly for the legal domain [14]. SaulLM-7B leverages the Mistral 7B architecture and extensive pretraining on a massive English legal corpus spanning over 30 billion tokens. The authors present a novel instructional finetuning method that further enhances SaulLM-7B’s performance on legal tasks. Notably, they also introduce LegalBench-Instruct, an improved evaluation protocol for assessing the legal proficiency of language models, building upon the LegalBench benchmark [23].

2.5.2. Legal Case Entailment

Legal case entailment is a crucial task in legal reasoning, requiring deep understanding of legal principles and precedents. Rosa et al. [57] made significant strides in this area by exploring fine-tuning techniques for BERT models to improve performance on legal case entailment tasks. Their research showed substantial improvements over baseline models, indicating that pre-trained language models can be effectively adapted to the nuances of legal language and reasoning. Building on this foundation, Goebel et al. [22] further investigated the use of LLMs for entailment in legal documents. Their work focused on two critical aspects of legal reasoning: summarizing legal arguments and identifying key legal principles.

2.5.3. Law Exams and Legal Question Answering

LLMs have shown remarkable performance in tackling law school exams and legal question answering tasks. Katz et al. [32] demonstrated that LLMs could achieve scores comparable to human law students on the Multistate Bar Examination (MBE), highlighting the potential of these models in legal education and assessment. Cui et al. [15] introduced Chatlaw, an innovative legal assistant that combines a Mixture-of-Experts (MoE) model with a multi-agent system to enhance the reliability and accuracy of AI-driven legal services. Their approach integrates knowledge graphs and artificial screening to construct a high-quality legal dataset for training. Addressing the need for a comprehensive overview of the field, Abdallah et al. [1] conducted an extensive survey on Legal Question Answering (LQA) systems. Their work provides valuable insights into the current state of LQA research, reviewing 14 benchmark datasets for question-answering in the legal field and presenting a comprehensive analysis of state-of-the-art Legal Question Answering deep learning models.

2.6. Retrieval Augmented Generation (RAG)

2.6.1. Overview of RAG

Retrieval-Augmented Generation (RAG) has emerged as a powerful paradigm in natural language processing, addressing key limitations of traditional large language models (LLMs). RAG systems enhance the capabilities of LLMs by incorporating external knowledge during the generation process, allowing for more accurate, up-to-date, and contextually relevant outputs.

The core principle of RAG, as introduced by Lewis et al. [38], is to augment the knowledge embedded in LLM parameters with information retrieved from external sources. This approach offers several advantages:

- Access to current information, overcoming the static nature of LLM training data
- Improved accuracy in domain-specific tasks
- Enhanced ability to provide citations and references for generated content
- Reduced hallucination and factual errors in model outputs

A typical RAG system consists of several key components:

1. A document corpus or knowledge base
2. An indexing and retrieval system
3. A large language model for generation
4. A mechanism to integrate retrieved information with the generation process

2.6.2. Recent Advancements in RAG

Subsequent research has focused on refining the RAG framework by enhancing the efficiency of the retrieval process and the integration of retrieved information into the generation pipeline. Innovations include improved methods for querying and retrieving relevant documents, as well as more effective ways to merge these findings with the initial queries to produce coherent and accurate final results[10, 29].

An advancement in the field of RAG was introduced by Guu et al. with their REALM (Retrieval-Augmented Language Model Pre-Training) framework [24]. This approach addressed a key limitation of previous RAG systems, which often functioned as black boxes due to the inherent lack of explainability in their generation models. REALM outlined a methodology for optimizing RAG across three critical stages: pre-training, fine-tuning, and inference. By integrating retrieval mechanisms into the pre-training process, REALM enabled the model to learn how to effectively utilize external knowledge from the outset.

This innovation not only improved the model's performance but also enhanced its interpretability. Additionally, their work made notable strides in unsupervised corpus alignment, providing a promising direction for future research in making RAG systems more transparent and adaptable to diverse knowledge sources.

Despite these advancements, RAG remains a vibrant area of ongoing research. Recent surveys by Li et al. and Gao et al. [39, 21] have synthesized the current state of the field, offering insights into the latest progress and categorizing the various specializations that have emerged. These categorizations help clarify the landscape of RAG research, illustrating the diverse approaches to enhancing model performance through external knowledge integration.

3 Related Work

3.1. Traditional Contract Review Methods

There have been many attempts in the field of contract review, or broadly speaking legal document analysis. The evolution of these methods can be traced from early automation attempts to more sophisticated approaches leveraging artificial intelligence.

3.1.1. Early automation attempts

Early efforts to automate contract review primarily relied on rule-based Natural Language Processing (NLP) techniques. These approaches involved predefined rule matching to extract clauses or relevant information from contracts. For instance, Al Qady and Kandil (2010), Liu et al. (2014), and Zhang and El-Gohary (2016) developed systems that used predefined vocabularies and rules to identify and extract specific contract elements [2, 40, 73].

Lee et al. (2019) proposed a rule-based automatic model to detect contract clauses that disadvantage contractors and extract critical information [36]. Their approach involved defining risk clauses, constructing a domain lexicon, and designing information extraction rules. Similarly, Lee et al. (2020) developed a rule-based NLP model for checking the omission of contractor-friendly clauses in contracts [37].

While these rule-based methods demonstrated impressive results in specific contexts, they faced limitations in scalability and generalization. The dependence on manually crafted rules made them less adaptable to varying contract types and formats, highlighting the need for more flexible approaches.

3.2. AI-powered Contract Analysis

The limitations of rule-based systems paved the way for more advanced AI-powered approaches to contract analysis. These methods leverage machine learning and, more recently, deep learning techniques to improve the flexibility and efficiency of contract review processes.

3.2.1. Machine learning approaches for contract review

Machine learning models have been widely applied to various aspects of contract analysis. Common algorithms include Support Vector Machines (SVM), Naïve Bayes (NB), k-Nearest Neighbors (KNN), and Hidden Markov Models (HMM). These models have been used for tasks such as clause classification, risk identification, and requirement extraction.

Hassan and Le (2020) developed a classification model using machine learning techniques that achieved a 95

3.2.2. Deep learning advancements

With the rise of deep learning, more sophisticated models have been applied to contract analysis. Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks have shown promise in this domain. For example, Hassan and Le (2021) used RNN to train a model for design-build (DB) requirement classification, highlighting the importance of text vectorization methods in classification reliability [26].

Choi and Lee (2022) proposed a bi-LSTM algorithm to train a risk level ranking (RLR) model for automatically analyzing key risk clauses in invitation to bid (ITB) documents at the bidding stage [13]. This approach demonstrates the potential of deep learning in more nuanced tasks such as risk assessment in contracts.

3.2.3. Challenges in data acquisition

Despite the advancements in ML and DL approaches, a significant challenge remains in acquiring high-quality training data. Construction contracts and other legal documents often contain sensitive corporate information, making it difficult to obtain large, diverse datasets for model training. This limitation has implications for the generalizability and robustness of AI models in contract review.

Table 3.1: Studies on NLP Methods for Contract Review

Method	Description	Task	Main Findings	Reference
Rule-based	Shallow parser for semantic knowledge extraction	Concept relation extraction	Proposed shallow parsing for semantic knowledge extraction	[2]
	Semantic and syntactic rules	Detrimental clause identification	Integrated rule-based NLP with domain knowledge	[36]
	NLP for SVO tagging	Hazardous clause extraction	Proposed semantic lexicon for risk clause identification	[33]
	Syntactic and semantic analysis	Contractor-friendly clause identification	Demonstrated rule-based NLP for missing clause identification	[37]
ML-based	Automating classification of text of requirements with NB, SVM, LR, kNN, DT, FNN	Contract requirements classification	Proposed automated framework using NLP and ML	[26]
	Semantic Analysis and Risk Level Ranking models	Risk clause classification	Integrated SA and RLR models for enhanced classification	[13]
	Various ML classifiers	Vague term identification	Combined NLP and ML for vague term detection	[12]
DL-based	Fine-tuned BERT	Safety report classification	Modified BERT for construction safety document classification	[19]
	Bi-LSTM	Named entity recognition	Developed Bi-LSTM for long-term dependencies in specifications	[47]
	BERT	Contract clause classification	Proposed using language models for clause classification	[46]
	Various LLMs	Contract text summarization	Developed merit-based evaluation for summarization	[69]

This table retains more detail from the original text, presenting key studies for each methodology (Rule-based NLP, ML-based, and DL-based). It includes descriptions of the methods used, the specific tasks addressed, main findings or proposals, and the corresponding references. The table structure allows for easy comparison between different approaches while maintaining most of the essential information from the original text.

3.3. LLMs in the Legal Domain

3.3.1. Contract Review and Analysis

The application of LLMs in contract review and analysis has emerged as a promising area of research, with the potential to significantly impact legal practice. As contracts form the backbone of many business and legal transactions, the ability to efficiently and accurately review these documents is of paramount importance. Recent advancements in LLM technology have shown remarkable progress in this domain, offering new possibilities for automating and enhancing the contract review process.

Martin et al. [43] conducted a comprehensive study that marked a significant milestone in this field. Their research compared the performance of advanced LLMs to human legal contract reviewers, including junior lawyers and legal process outsourcers (LPOs). The findings were striking: the LLMs demonstrated the ability to match or even exceed human accuracy in identifying legal issues within contracts. Moreover, the AI-driven approach dramatically reduced both the time and cost associated with contract review. This study highlights the potential for LLMs to revolutionize the efficiency of legal practices, particularly in areas that involve high volumes of contract processing.

Demonstrating the versatility of LLMs in specialized contract domains, Wong et al. [68] proposed a novel approach to construction contract risk identification. Their method incorporates domain-specific knowledge into LLMs, resulting in improved performance in recognizing risk clauses within construction contracts. This research underscores the adaptability of LLMs to specific legal niches and highlights the potential for these models to be fine-tuned for various specialized areas of contract law.

3.3.2. Fine-Tuned Language Models for the Legal Domain

Building on the success of general-purpose LLMs, there is a growing interest in fine-tuning these models for specific legal tasks to further enhance their effectiveness. Fine-tuned models offer another promising avenue for improving performance on specialized legal tasks, complementing the capabilities of general-purpose models.

An exemplary case is SaulLM-7B, a pioneering 7 billion parameter large language model tailored explicitly for the legal domain [14]. As the first publicly available LLM designed for legal text comprehension and generation, SaulLM-7B leverages the Mistral 7B architecture and extensive pretraining on a massive English legal corpus spanning over 30 billion tokens. The authors present a novel instructional finetuning method that further enhances SaulLM-7B's performance on legal tasks by leveraging additional legal datasets. SaulLM-7B exhibits state-of-the-art proficiency in understanding and processing legal documents, outperforming existing generic models. Notably, the authors also introduce LegalBench-Instruct, an improved evaluation protocol for assessing the legal proficiency of language models, building upon the LegalBench benchmark[23]. By incorporating legal tasks from the MMLU benchmark, focusing on areas such as international law and jurisprudence, LegalBench-Instruct provides a comprehensive framework for evaluating and refining legal LLMs.

3.3.3. Challenges and Limitations of LLMs in Legal Tech

Despite the impressive capabilities of LLMs in legal tasks, several challenges and limitations need to be addressed:

- Hallucination: LLMs can generate information that is incorrect, misleading, or entirely fabricated, a phenomenon known as hallucination [44, 28]. This is particularly problematic in the legal domain, where accuracy and reliability are paramount.
- Domain-specific knowledge: While LLMs have vast general knowledge, they may lack the depth of understanding required for specialized legal tasks. This limitation necessitates the development of domain-specific models or the augmentation of general LLMs with legal knowledge.
- Interpretability and explainability: The ”black box” nature of LLMs poses challenges in legal applications where transparency and explainability are crucial. Developing methods to provide clear reasoning for LLM outputs in legal contexts remains an important area of research.
- Ethical and privacy concerns: The use of LLMs in processing sensitive legal documents raises important ethical and privacy considerations that must be carefully addressed.

3.4. Integration of Knowledge Graphs and RAG

3.4.1. Knowledge Graph with LLMs

The integration of these knowledge graphs with recent advancements in large language models has demonstrated significant progress in enhancing information retrieval, reasoning, and question answering capabilities across various domains [31, 60, 41, 71, 67, 54]. Large language models can leverage the structured data and factual knowledge provided by knowledge graphs to improve their performance on tasks like factual reasoning, domain-specific question answering, and knowledge-intensive applications. These developments underscore the increasing importance of knowledge graphs in advancing artificial intelligence systems.

For graph-based reasoning, the works of Think-on-Graph [60] and Reasoning-on-Graph [41] have enhanced large language models' reasoning abilities by directly integrating knowledge graphs. These approaches allow the language models to leverage the structured knowledge and relational information stored in the graphs to improve their logical inference and reasoning performance.

In the realm of factual reasoning, Yang et al. [71] propose augmenting large language models like ChatGPT with knowledge graphs across various training phases. By incorporating factual knowledge from the graphs, the language models can enhance their capabilities in answering questions that require accurate retrieval and synthesis of information.

For domain-specific question answering, Wen et al.'s Mindmap [67] and Qi et al.'s FoodGPT [54] leverage knowledge graphs to boost the inference capabilities of large language models. In these works, the knowledge graphs provide the models with structured information about specialized domains, such as medicine and food, enabling them to better understand and respond to queries within those contexts.

Overall, these contributions underscore the increasing efficacy of integrating large language models with knowledge graphs to improve information retrieval, reasoning, and question answering across a variety of applications and domains. The structured data and factual knowledge provided by the knowledge graphs complement the language modeling and generation capabilities of the large language models, leading to enhanced performance and more intelligent outcomes.

3.5. Optimizing the RAG System

This section provides an in-depth discussion of the Retrieval-Augmented Generation (RAG) system, focusing on various optimization techniques applied across different stages to improve overall performance. A typical RAG pipeline consists of the following stages:

1. **Indexing**: Documents are split into chunks and used as input for embedding models to generate vector representations, enabling efficient search.
2. **Retrieval**: The query is transformed into a vector using the same embedding model applied during indexing, which is then used to search the indexed database to identify the most relevant document chunks.
3. **Generation**: The final stage involves generating responses based on the retrieved document chunks, typically using a generative model like GPT-4 to synthesize a coherent and contextually relevant response.

3.5.1. Indexing

The indexing stage is foundational to the RAG pipeline, converting documents into a format that allows for efficient and accurate retrieval. This process includes two main tasks: document chunking and embedding generation.

Chunking Optimization

Chunking determines how documents are divided and organized within the vector database, directly influencing retrieval outcomes and, consequently, the final generated outputs [58]. Given the diverse nature of documents, different chunking strategies may be employed to provide suitable and contextually rich content for retrieval.

One common technique is recursive chunking, or the sliding window method, where documents are

Stage	Technique	Advantages	Additional Info
Indexing	Recursive Chunking	Maintains context connectivity	Sliding window method
	Semantic Chunking	Identifies coherent segments	Uses NLP (NLTK, spaCy)
	Hybrid Chunking	Combines multiple methods	Recursive + semantic
	Customized Chunking	Tailored for specific docs	E.g., Financial report[72]
	Multi-vector Indexing	Captures multiple aspects	For specialized domains
Retrieval	Query Expansion	Broadens scope, enhances recall	Synonym, contextual, KG-based
	Hierarchical Retrieval	Leverages doc structure	For structured docs
	Query Routing	Improves efficiency & relevance	Topic-based, dynamic selection
	Adaptive K Selection	Balances recall & efficiency	Dynamic K, confidence-based
	Hybrid Retrieval	Combines vector & traditional IR	BM25-augmented, two-stage
	Reranking	Refines relevance & diversity	Cohere, FlashRank[16], G-RAG[18]
Generation	Fine-tuning	Enhances domain performance	Adapts to domain data
	Post-processing	Improves accuracy & coherence	Re-ranking, filtering, grammar
	Feedback Loops	Enables continuous improvement	User or automated metrics

Table 3.2: RAG Improvement Techniques

split into chunks of relatively similar size while preserving contextual connectivity. This method involves initially dividing the document into larger chunks based on predefined boundaries (e.g., section headers or paragraph breaks) and then recursively splitting these into smaller chunks until they meet a desired size limit.

Alternatively, semantic chunking utilizes natural language processing (NLP) techniques to identify semantically coherent segments within a text. NLP libraries like NLTK or spaCy provide tools for tokenization, part-of-speech tagging, named entity recognition, and other linguistic analyses that can help identify meaningful chunks based on the text's semantic content.

Hybrid approaches may also be employed, combining recursive chunking to maintain document structure with semantic chunking for greater coherence and relevance. For example, recursive chunking could first divide documents into large sections, followed by semantic chunking to refine these sections further based on their content.

An example of a customized chunking strategy is proposed by Yepes et al. for financial reports [72]. They use a combination of token-based chunking and element-based chunking techniques, supported by computer vision and NLP tools, to extract meaningful elements (e.g., titles, texts, tables) and merge them into coherent chunks. This approach incorporates metadata, representative keywords, and summaries, which enhances retrieval and generation processes by capturing both structural and semantic information unique to financial documents.

Multi-Vector Indexing

To further enhance retrieval, multi-vector indexing techniques can be employed. Rather than representing each document or chunk with a single dense vector, multiple vector embeddings are used to capture different aspects or views of the text, such as semantic content, document structure, or metadata. During retrieval, these multiple vectors are combined using methods like late interaction or multi-vector attention, which improves the matching of query representations to more accurately surface relevant chunks. This approach is particularly beneficial for domains with specialized vocabularies or complex structures, such as legal or technical documents.

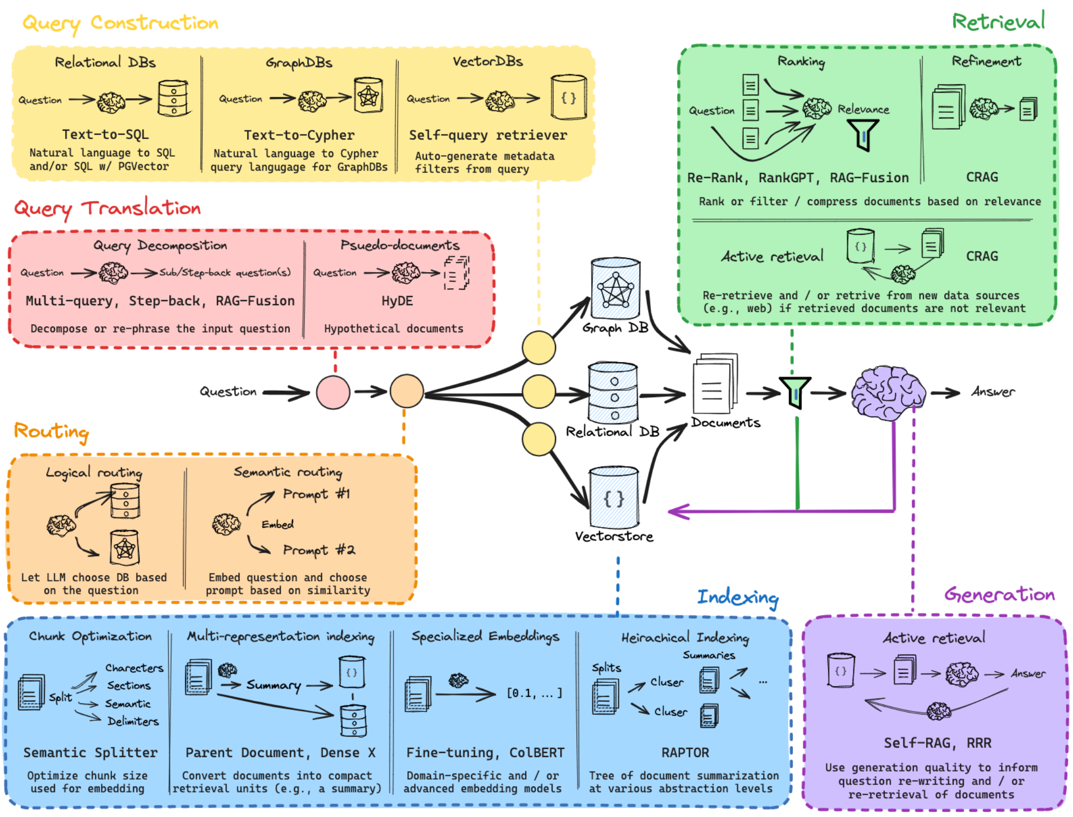

Query Construction
Relational DBs
Question
Text-to-SQL
Natural language to SQL and/or SQL w/ PGVector
GraphDBs
Question
Text-to-Cypher
Natural language to Cypher query langugage for GraphDBs
VectorDBs
Question
Self-query retriever
Auto-generate metadata filters from query
Retrieval
Ranking
Question
Relevance
Re-Rank, RankGPT, RAG-Fusion
Refinement
CRAG
Rank or filter / compress documents based on relevance
Active retieval
CRAG
Re-retrieve and / or retrive from new data sources (e.g., web) if retrieved documents are not relevant
Query Translation
Query Decomposition
Question
Sub/Step-back question(s)
Multi-query, Step-back, RAG-Fusion
Decompose or re-phrase the input question
Psuedo-documents
Question
HyDE
Hypothetical documents
Question
Graph DB
Relational DB
Vectorstore
Documents
Answer
Routing
Logical routing
Let LLM choose DB based on the question
Semantic routing
Prompt #1
Embed
Prompt #2
Embed question and choose prompt based on similarity
Indexing
Chunk Optimization
Split
Charecters
Sections
Semantic
Delimiters
Semantic Splitter
Optimize chunk size used for embedding
Multi-representation indexing
Summary
Parent Document, Dense X
Convert documents into compact retrieval units (e.g., a summary)
Specialized Embeddings
[0.1, ...]
Fine-tuning, ColBERT
Domain-specific and / or advanced embedding models
Heirachical Indexing
Summaries
Splits
Cluser
Cluser
...
RAPTOR
Tree of document summarization at various abstraction levels
Generation
Active retieval
Answer
Self-RAG, RRR
Use generation quality to inform question re-writing and / or re-retrieval of documents

Figure 3.1: Various Techniques used in RAG system, from LangChain [35]

Embedding Models

Embedding models are crucial in transforming textual data into dense vector representations, enabling efficient similarity computations between queries and documents. In a RAG system, documents or chunks are first converted into embeddings using pre-trained models, such as BERT or RoBERTa, and stored in a vector database for fast approximate nearest neighbor search. When a query is submitted, it is similarly converted into an embedding, and the most relevant document embeddings are retrieved based on their similarity to the query.

Muennighoff et al. introduce the Massive Text Embedding Benchmark (MTEB) to address the limitations of existing benchmarks that often focus narrowly on specific tasks such as semantic textual similarity (STS) [48]. MTEB spans 165 datasets across 113 languages, covering eight embedding tasks. Their findings highlight the lack of a single embedding method that consistently performs well across all tasks, suggesting the need for continued exploration of diverse approaches in text embedding.

3.5.2. Retrieval

The retrieval stage connects user queries with the corpus of indexed documents, identifying and retrieving the most relevant chunks to support accurate and comprehensive responses. This stage is critical to the overall quality and relevance of the output.

Many techniques from traditional Information Retrieval (IR) can be adapted for RAG retrieval. For example, traditional relevance metrics can be adapted to the dense vector space, or query expansion techniques can enhance recall by enriching the initial query with related terms or concepts. Incorporating user feedback or pseudo-relevance feedback can further refine retrieval results.

Pre-Retrieval Optimization

The pre-retrieval stage focuses on optimizing queries to improve the efficiency and accuracy of retrieval. Techniques such as query expansion augment the original query with additional terms or concepts to broaden the search scope. Approaches include synonym expansion, contextual expansion using language models, and knowledge graph-based expansion to identify related entities.

Hierarchical retrieval techniques are also explored, leveraging structural information in documents to index content at multiple granularity levels. This hierarchical process enables more focused and accurate retrieval by narrowing down the search space before selecting the most relevant chunks.

Query routing further enhances retrieval efficiency by directing queries to the most relevant subset of documents. Techniques such as topic-based routing, hierarchical indexing, and dynamic index selection are employed to refine the retrieval process.

Retrieval Algorithms

Most vector databases utilize similarity search based on query and document vectors. Optimization involves selecting the appropriate similarity metric, such as cosine similarity, Euclidean distance, or dot product, depending on the embedding type. Adaptive techniques for determining the optimal number of neighbors (K) to retrieve, and combining vector-based retrieval with traditional IR methods, such as BM25, can enhance performance.

Post-Retrieval Refinement

Post-retrieval techniques, such as reranking, aim to improve the relevance and diversity of the final set of retrieved documents. Advanced reranking models like Cohere's reranker[1], FlashRank [16], and G-RAG [18], which leverage graph neural networks, can enhance retrieval quality by incorporating contextual information and document connections.

3.5.3. Generation

The generation stage synthesizes information from the retrieved document chunks to produce coherent and contextually relevant responses. This stage relies heavily on the capabilities of generative models, such as GPT-4, which are fine-tuned on domain-specific data to generate accurate responses.

Post-Processing Techniques

Post-processing techniques can refine the generated output, ensuring relevance, accuracy, and coherence. Methods include re-ranking responses, filtering out irrelevant or incorrect information, and applying grammatical corrections.

Feedback Loops

Feedback loops, whether from direct user feedback or automated evaluation metrics, enable continuous improvement of the generative model. This adaptive process ensures that the model becomes more effective over time.

[1] https://cohere.com/rerank

4 A Revisit of Contract Review

4.1. Problem Definition

In practical scenarios, contract review often involves a comprehensive checklist of items requiring scrutiny. This checklist encompasses both fundamental information, such as the contract's title, involved parties, and effective date, as well as more nuanced elements, including post-termination or post-expiration obligations for any party. The review process necessitates a systematic examination of the contract, addressing each checklist item sequentially.

4.1.1. Example Use Case

The Overwhelmed Small Business Owner

Alex runs a small tech startup that's been gaining traction lately. With success comes a flood of contracts, agreements, and legal documents that need careful review. Alex, like many entrepreneurs, didn't start a business to spend countless hours deciphering legal jargon. Yet, here they are, night after night, poring over dense texts, trying to spot any clause that might put the company at risk.

One evening, while reviewing a particularly complex supplier agreement, Alex misses a crucial detail about intellectual property rights. This oversight nearly costs the company a significant amount of money and potential future earnings. Realizing the gravity of the situation, Alex considers hiring a lawyer for every contract review, but the startup's budget is already stretched thin. What Alex really needs is **a tool that can quickly scan these documents, highlight the important parts, and flag any potential issues. Something that could give a clear summary of key points like payment terms, liabilities, and termination conditions.** With such a tool, Alex could focus on growing the business instead of drowning in legal documents, and only call in expensive legal help when absolutely necessary.

4.1.2. Mathematical Formulation

We formulate this problem as an extension of classical information retrieval (IR), incorporating additional complexities inherent to contract analysis. Let $D = \{d_1, d_2, ..., d_n\}$ represent the set of document chunks from a given contract, where each d_i is a textual segment. Let $Q = \{q_1, q_2, ..., q_m\}$ denote the set of queries derived from the checklist items. The primary task can be formalized as follows:

For each query $q_j \in Q$, find the most relevant chunk $d_i \in D$ such that:

$$d_i^*(q_j) = \arg\max_{d_i \in D} \text{relevance}(q_j, d_i) \tag{4.1}$$

Where relevance(q_j, d_i) is a function that measures the semantic similarity or relevance between the query q_j and the document chunk d_i.

It is important to note that not all checklist items may be applicable or present within a given contract, resulting in potential null or ”not applicable” (N/A) responses. This can be represented as:

$$\text{response}(q_j) = \begin{cases} \text{extract}(d_i^*) & \text{if relevance}(q_j, d_i^*) > \theta \\ \text{N/A} & \text{otherwise} \end{cases} \tag{4.2}$$

Where θ is a predefined relevance threshold, and extract$(\cdot)$ is a function that extracts the pertinent information from the most relevant chunk.

A critical consideration in this context is that the ground truth typically resides within a single chunk of the document. Consequently, if the retrieval system fails to identify and extract the correct chunk, it becomes impossible for the subsequent processing to generate an accurate output. Formally, let d_t be the chunk containing the ground truth for query q_j. The system's performance is highly dependent on:

$$P(d_i^* = d_t | q_j, D) \tag{4.3}$$

This high-stakes scenario underscores the significant impact of incorrect retrieval results, emphasizing the paramount importance of maximizing the hit rate within the retrieval system.

While the foundation of our problem lies in information retrieval, several key factors distinguish it from traditional IR tasks:

1. **Post-retrieval Processing:** Unlike traditional IR, our system employs advanced language models to process and synthesize the retrieved information:

$$\text{output}(q_j) = \text{LM}(\text{response}(q_j), q_j) \tag{4.4}$$

   Where LM$(\cdot)$ represents the language model's processing function.

2. **Null Response Handling:** The system must determine when a query is not applicable to the given contract:

$$\text{output}(q_j) = \begin{cases} \text{LM}(\text{response}(q_j), q_j) & \text{if response}(q_j) \neq \text{N/A} \\ \text{null} & \text{otherwise} \end{cases} \tag{4.5}$$

Our primary focus remains on optimizing the retrieval process to maximize the probability of correctly identifying the relevant document chunk. To address this challenge, our main contribution lies in enhancing the context representation and input processing. By applying our proposed techniques to both the context and input before generation, we aim to improve the overall quality of the output. This approach is formulated as:

$$\text{output}(q_j) = \text{LM}(\text{enhance}(\text{response}(q_j)), \text{enhance}(q_j)) \tag{4.6}$$

Where enhance$(\cdot)$ represents our novel preprocessing function applied to both the retrieved response and the original query. This enhancement aims to provide richer context and more informative input to the language model, potentially leading to more accurate and contextually appropriate outputs.

4.2. Intuitive Solution: Naive RAG System

The intuitive approach to addressing this problem is to implement a basic Retrieval-Augmented Generation (RAG) system. This naive solution typically consists of three main components:

1. **Indexing:** The contract document is split into chunks and indexed in a vector database.
2. **Retrieval:** Given a query, the system retrieves the most relevant chunks based on similarity scores.
3. **Generation:** A language model generates a response based on the retrieved chunks and the query.

While this approach provides a functional baseline, it suffers from several limitations:

1. **Contextual Insensitivity:** Simple chunking may break important contextual relationships within the document.
2. **Retrieval Accuracy:** The system may fail to retrieve the most relevant chunks, especially for complex or ambiguous queries.
3. **Limited Query Understanding:** The naive approach doesn't account for potential mismatches between query language and document terminology.
4. **Lack of Structural Information:** The flat representation in vector databases doesn't capture the hierarchical nature of contracts.
5. **Inefficient Null Response Handling:** The system may struggle to efficiently determine when a query is not applicable to the given contract.

These drawbacks highlight the need for a more sophisticated approach that can better handle the complexities of contract analysis.

4.3. Proposed Solution: Enhanced RAG System

To address the limitations of the naive approach, we propose an enhanced RAG system that incorporates several optimizations:

1. **Chunking Optimization:** We employ advanced techniques to preserve contextual relationships and semantic coherence when splitting the document.
2. **Query Expansion:** We enhance queries with relevant terms and synonyms to improve retrieval accuracy.
3. **Chunk Reranking:** After initial retrieval, we apply a reranking step to further refine the selection of relevant chunks.
4. **Context and Input Enhancement:** We apply preprocessing techniques to both the retrieved context and the original query to provide richer information to the language model.

These enhancements aim to significantly improve the system's ability to retrieve the correct and matching context for each query, thereby increasing the likelihood of generating accurate and contextually appropriate responses.

The proposed approach represents a novel and useful contribution to the field of contract analysis and legal document processing for several reasons:

1. **Improved Contextual Understanding:** By optimizing the chunking process and implementing query expansion, our system demonstrates a more nuanced understanding of complex legal documents. This is particularly crucial in the legal domain, where context and precise interpretation are paramount.
2. **Enhanced Retrieval Accuracy:** The combination of advanced retrieval mechanisms, including pre-retrieval optimization and post-retrieval refinement, addresses a critical challenge in legal document analysis - the accurate identification of relevant information. This is especially valuable given the often lengthy and intricate nature of legal contracts.
3. **Scalability and Flexibility:** Our modular architecture allows for easy integration of different components, such as embedding models or language models. This flexibility makes the system adaptable to various legal contexts and evolving technological advancements, ensuring its long-term relevance and applicability.
4. **Balanced Approach to Precision and Recall:** By implementing a multi-step retrieval process with both expansion and reranking, we strike a balance between casting a wide net for potentially relevant information and precisely identifying the most pertinent content. This is crucial in legal applications where both comprehensiveness and accuracy are essential.
5. **Tailored for Legal Domain:** The system's design, particularly the prompt engineering aspect, is specifically tailored for legal document analysis. This domain-specific approach addresses the

unique challenges of legal text processing, such as the need for precise interpretation and the critical importance of identifying specific clauses or terms.

The novelty of our approach lies not just in the individual components, but in their thoughtful integration and optimization for the legal domain. By addressing the specific challenges of contract analysis, our system offers a more reliable and efficient tool for legal professionals, potentially reducing the time and resources required for contract review and interpretation.

Furthermore, this research contributes to the broader field of natural language processing by demonstrating how domain-specific knowledge and requirements can be effectively incorporated into a general-purpose RAG framework. The insights gained from this study could inform the development of similar systems in other specialized fields that deal with complex, context-dependent document analysis.

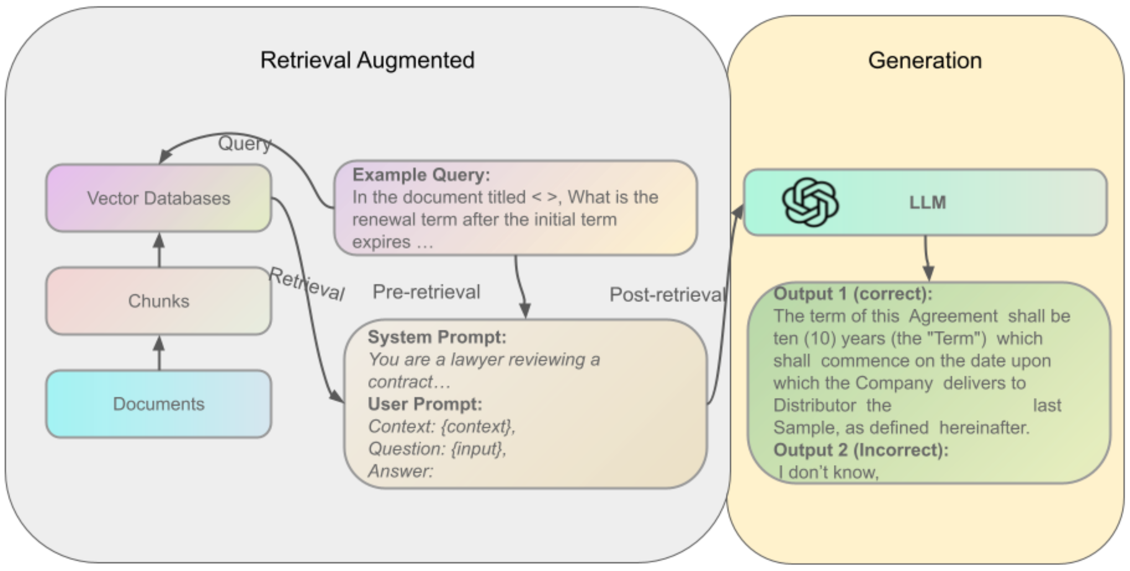

Retrieval Augmented
Generation
Query
Vector Databases
Example Query:
In the document titled < >, What is the renewal term after the initial term expires …
LLM
Chunks
Retrieval
Pre-retrieval
Post-retrieval
System Prompt:
You are a lawyer reviewing a contract…
User Prompt:
Context: {context},
Question: {input},
Answer:
Documents
Output 1 (correct):
The term of this Agreement shall be ten (10) years (the "Term") which shall commence on the date upon which the Company delivers to Distributor the last Sample, as defined hereinafter.
Output 2 (Incorrect):
I don't know,

Figure 4.1: Architectural overview of the proposed RAG system

4.3.1. Indexing

Our system primarily operates on textual data. For the CUAD [27] and ContractNLI [34] datasets, we utilize the provided text transcriptions of various PDF files. To accommodate new incoming documents in raw formats, we have implemented parsing functions to convert them into plain text, ensuring compatibility with our pipeline.

Chunking

Document chunking is performed using a recursive text splitter, with chunks set to 2048 characters. This configuration results in average token lengths of approximately 300-400, aligning with recommendations from Microsoft's GraphRAG. While this approach serves as our default, we plan to explore alternative splitting strategies, such as naive character-based or paragraph-based splitting, as part of our comparative experiments.

4.3.2. Embedding

For text embedding, we employ OpenAI's default embedding models. This choice is motivated by their widespread adoption and accessibility. Given that our research utilizes public datasets, there are no immediate privacy concerns regarding the test data. However, the system architecture allows for seamless integration of open-source embedding models if required for specific use cases or privacy considerations.

4.3.3. Retrieval Mechanisms

Retrieval Algorithm

Our retrieval mechanism leverages vector similarities with a threshold of 0.6 and a retrieval limit of K=3 chunks. This configuration optimizes the retrieval of relevant information while maintaining a manageable context size for the language model. Although most queries typically correspond to a single correct chunk, this approach provides a buffer to enhance recall.

Storage Solutions

We employ two primary storage solutions: Chroma[1] serves as our default vector database.

Pre-retrieval Optimization

To enhance retrieval accuracy, we implement query expansion and routing techniques. Furthermore, we utilize keyword matching to prefilter chunks from relevant documents, thereby improving the precision of our retrieval process. These methods collectively contribute to refining our information retrieval capabilities.

Post-retrieval Refinement

We incorporate a reranking step using FlashRank with a cross-encoder as the default configuration. This post-retrieval refinement aims to further improve the relevance of retrieved chunks before they are passed to the generation phase.

4.3.4. Generation

Prompt Engineering

Our system employs a carefully crafted prompt to guide the language model:

```
system_prompt = """You are a lawyer reviewing a contract.
When given a question, answer based on the information in the contract.
If asked about specific information, either provide it or state 'I don't know'.
Use the provided context, but be aware it may not always be relevant.
Be honest and provide correct information.
Provide answers as they appear in the document, minimizing additional information.
"""
prompt = ChatPromptTemplate.from_messages([
SystemMessage(system_prompt),
HumanMessagePromptTemplate.from_template(
"Context: {context}\nQuestion: {input}\nAnswer:")
])
```

This prompt structure aims to elicit precise, contract-specific responses while encouraging the model to acknowledge uncertainty when appropriate.

Language Model Integration

We utilize `gpt-4o-2024-05-13` as our primary generation model, balancing advanced capabilities with cost-effectiveness. This choice allows us to maintain high-quality outputs while minimizing potential negative influences on our final results. The modular nature of our system architecture allows for easy substitution with alternative language models as needed for specific research objectives or performance comparisons.

4.4. Comparative Advantage in Contract Review

To fully appreciate the efficacy of our enhanced RAG system for contract review, it is instructive to compare it both to the traditional human review process and to a plain RAG framework. This comparison elucidates the specific advantages our system offers in the context of legal document analysis.

[1] https://www.trychroma.com/

4.4.1. Comparison with Human Contract Review

The human process of reviewing a contract is a multi-faceted and complex task that requires both attention to detail and a broad understanding of legal principles. Typically, a lawyer begins by quickly scanning the document to grasp its overall structure and main sections. This initial skim is followed by a detailed reading, where the lawyer carefully examines each clause, paying close attention to specific legal terminology and potential implications. Throughout this process, the lawyer cross-references important clauses with other parts of the document or external legal standards, ensuring a comprehensive understanding of the contract's interconnected elements.

As the review progresses, the lawyer identifies key issues that require special attention or negotiation. This step relies heavily on the lawyer's expertise and experience, allowing them to recognize potential problems or advantageous clauses that might not be immediately apparent. The final stages involve a deep analysis and interpretation of the contract's contents, where the lawyer considers the implications of various clauses and their interrelationships. Finally, the lawyer documents their findings, concerns, and recommendations for further action or discussion.

Our enhanced RAG system emulates and augments several aspects of this human process. The chunking optimization mimics the human ability to understand document structure and context, similar to the initial skim and detailed reading phases. By preserving semantic coherence, our system can maintain the integrity of complex legal clauses that might span multiple paragraphs, a crucial factor in accurate interpretation.

The query expansion and reranking features of our system parallel the cross-referencing step in human review. These capabilities ensure that relevant information is retrieved even when not explicitly mentioned in the query, mirroring a lawyer's ability to connect related concepts and clauses. Furthermore, our system's capacity to quickly process and analyze large volumes of text surpasses human capabilities in terms of speed and consistency, particularly in identifying key issues across multiple documents.

While human expertise remains crucial for nuanced interpretation and strategic decision-making, our system provides rapid, consistent responses to specific queries, supporting the analysis phase of contract review. This synergy between AI capabilities and human expertise creates a powerful tool for enhancing the efficiency and accuracy of contract analysis.

4.4.2. Advantages over Plain RAG Framework

When compared to a plain RAG framework, our enhanced system offers several key advantages in the context of contract review. The most significant of these is the preservation of contextual integrity. Our advanced chunking techniques ensure that the semantic coherence of contract sections is maintained, which is crucial for understanding complex legal clauses. A plain RAG system, by contrast, might arbitrarily split such clauses, potentially leading to misinterpretation of critical contract elements.

Another major advantage lies in our system's handling of legal terminology. The query expansion feature is particularly beneficial for legal documents, where synonyms and related legal concepts play a crucial role. Our system can capture variations in legal terminology that a plain RAG system might miss, ensuring a more comprehensive retrieval of relevant information.

The reranking step in our system addresses a critical need in legal document analysis: precision. In the legal field, where the distinction between similar concepts can have significant implications, our system's ability to differentiate between nuanced legal terms is invaluable. This feature helps in distinguishing between similar but distinct legal concepts, which a plain RAG system might conflate based on surface-level similarity.

Furthermore, our system's context and input enhancement techniques are specifically designed to mimic legal reasoning patterns. This results in a more nuanced understanding of legal documents compared to a generic RAG framework. By providing more accurate and relevant context to the language model, our system also reduces the likelihood of generating incorrect or irrelevant information, a critical factor in legal document analysis where accuracy is paramount.

4.4.3. Practical Implications

The enhanced capabilities of our system translate into several practical benefits for contract review. Perhaps the most immediate impact is the significant increase in efficiency. By automating the initial stages of contract review, our system can substantially reduce the time lawyers spend on routine aspects of document analysis. This allows legal professionals to focus their expertise on more complex issues and strategic decision-making.

Consistency is another key advantage of our system. It ensures a uniform approach to contract analysis across different documents and reviewers, reducing the risk of oversight or inconsistent interpretation. This is particularly valuable in large organizations or law firms where multiple individuals may be involved in reviewing related documents.

The accuracy of contract review is also enhanced by our system. By leveraging advanced NLP techniques tailored for legal documents, it can identify subtle details and potential issues that might be overlooked in a manual review or when using a generic RAG implementation. This capability is especially useful in complex contracts where small details can have significant legal or financial implications.

Scalability is a significant benefit of our system, particularly in the context of large-scale due diligence processes or compliance reviews. The system can handle large volumes of contracts or complex, multi-document agreements more efficiently than human reviewers or basic RAG systems. This scalability makes it an invaluable tool for law firms and corporate legal departments dealing with high volumes of contracts.

It's important to note that our enhanced RAG system is not designed to replace human legal experts, but rather to augment their capabilities. By handling the more routine and time-consuming aspects of contract review, the system allows legal professionals to focus on high-level analysis, strategic decision-making, and client interaction. This synergy between advanced NLP techniques and legal domain knowledge represents a significant step forward in the application of AI to complex document analysis tasks in the legal field.

In conclusion, while our enhanced RAG system does not replicate the full spectrum of human legal expertise, it offers a powerful complement to human review processes. It addresses many of the limitations of plain RAG frameworks in the specific context of contract analysis, providing a more nuanced, accurate, and efficient tool for legal professionals. As the legal industry continues to embrace technological solutions, systems like ours are poised to play an increasingly important role in enhancing the quality and efficiency of legal services.

5 Knowledge Graph-based RAG System: An Advanced Approach

In Chapter 4, we introduced our enhanced RAG system and compared it with the conventional approach. We now propose a further refinement: replacing the vector database with a graph database, specifically a Knowledge Graph (KG). This chapter explores how this innovative approach addresses some of the limitations of traditional RAG systems and offers new possibilities for contract analysis. As discussed in Section 2.4, a Knowledge Graph offers promising potential to structure unstructured contract files, transforming them into traceable and interconnected nodes. This enhancement is motivated by two primary factors:

1. Traditional RAG systems heavily rely on similarity search, often failing to capture contextual connections between adjacent chunks and ultimately depending on the sensitivity of LLMs. By employing a KG, we can easily extract contextual information and provide relationship data as comprehensive context to the LLM
2. KGs allow for the creation of additional relationships and nodes to further organize our data (which represent text chunks in the vector store). This approach enables us to reconstruct unstructured contract documents into a book-style graph, complete with a ”table of contents” that facilitates navigation to desired sections, as opposed to indiscriminately searching across multiple pages.

The use of a Knowledge Graph in our RAG system represents a significant departure from traditional approaches. It allows us to capture and utilize the inherent structure and relationships within contract documents, which are often lost in simple chunk-based vector stores. This structured approach is particularly valuable in the legal domain, where the context and relationships between different parts of a document are crucial for accurate interpretation.

For example, in a traditional RAG system, a query about a specific clause might return relevant chunks based on keyword similarity, but it might miss important context from related clauses or definitions elsewhere in the document. Our KG-based approach, however, can navigate these relationships, providing a more comprehensive and accurate response.

While reasoning on KGs has historically been challenging due to the complexity of determining an appropriate starting point within an intricate graph, we propose a compromise. By utilizing the similarity search functionality of traditional RAG systems, we can establish a suitable starting point, which, while not always precise, is generally sufficient for our purposes. This hybrid approach combines the strengths of both vector-based similarity search and graph-based relational data, offering a more robust solution for complex document analysis.

5.1. Knowledge Graph Construction

The construction of our knowledge graph is a critical process that forms the foundation of our enhanced RAG system. This section details the steps involved in transforming unstructured contract documents into a structured, navigable graph. We utilize Neo4J as our graph database platform due to its robustness and flexibility in handling complex graph structures.

5.1.1. Creating Text Nodes

The creation of text nodes is the first and most fundamental step in our knowledge graph construction process. These nodes serve as the basic units of information in our graph, representing chunks of text from the original documents. This process involves the following steps:

1. **Document Segmentation:** We divide each document into chunks of 2048 characters. This size was chosen as a balance between granularity and context preservation.
2. **Metadata Augmentation:** Each chunk is enhanced with metadata to provide context and facilitate navigation. The metadata includes:
   - `seqId`: The index of the chunk within the document (starting from 0)
   - `section`: Index of the section
   - `text`: The actual content of the chunk
   - `chunkId`: A unique identifier combining `seqId` and `source`
   - `source`: Name of the source document
   - `names`: Related company names
3. **Embedding Calculation:** We compute embeddings for each chunk using OpenAI's `text-embedding-3-small` model[1]. This results in a 1536-dimensional vector representation of the text, which we store as the `embedding` property.
4. **Database Insertion:** The resulting data is inserted into our Neo4J database as `Chunk` nodes.

Figure 5.1 illustrates the structure of a `Chunk` node in our knowledge graph.

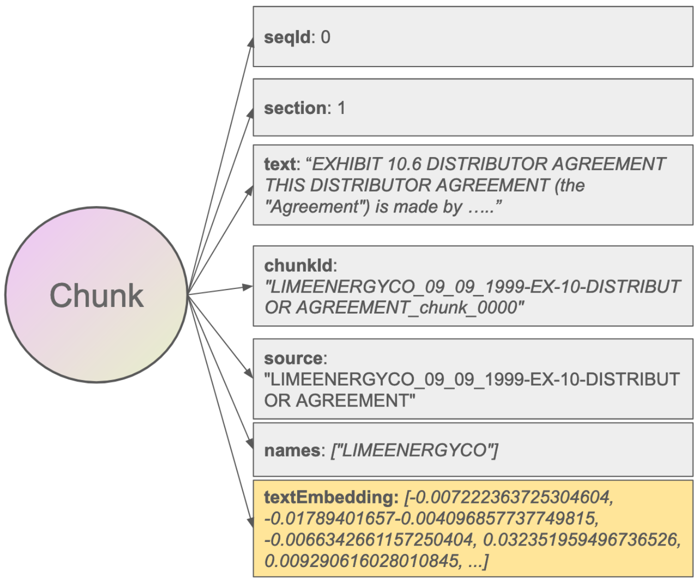

Chunk
seqId: 0
section: 1
text: "EXHIBIT 10.6 DISTRIBUTOR AGREEMENT THIS DISTRIBUTOR AGREEMENT (the "Agreement") is made by ….."
chunkId: "LIMEENERGYCO_09_09_1999-EX-10-DISTRIBUTOR AGREEMENT_chunk_0000"
source: "LIMEENERGYCO_09_09_1999-EX-10-DISTRIBUTOR AGREEMENT"
names: ["LIMEENERGYCO"]
textEmbedding: [-0.007222363725304604, -0.01789401657-0.004096857737749815, -0.0066342661157250404, 0.032351959496736526, 0.009290616028010845, ...]

Figure 5.1: Structure of a Chunk node in the knowledge graph

To maintain data integrity and prevent duplicate entries, we implement a uniqueness constraint in Neo4J:

```
CREATE CONSTRAINT unique_chunk IF NOT EXISTS
FOR (c:Chunk) REQUIRE c.chunkId IS UNIQUE
```

[1] https://platform.openai.com/docs/guides/embeddings/embedding-models

This constraint ensures that each `Chunk` node in our graph has a unique `chunkId`, preventing any potential data inconsistencies.

5.1.2. Adding Relationships

The addition of relationships to our knowledge graph is what truly sets it apart from a simple collection of text chunks. These relationships capture the structure and context of the original documents, allowing for more sophisticated querying and analysis.

To enhance the utility of our knowledge graph, we introduce two types of relationships:

1. **NEXT:** This relationship maintains the sequential order among nodes within individual documents. It creates a linked-list style structure, allowing for easy navigation between adjacent chunks.
2. **PART_OF:** This relationship links chunk nodes to their respective document nodes, establishing a hierarchical structure within our graph.

The process of establishing these relationships involves the following steps:

1. We group text nodes by their `source` attribute.
2. Within each group, we sort the nodes by their `seqId`.
3. We iterate through the sorted nodes, creating a `NEXT` relationship between each node and its successor.
4. We create a `Document` node for each unique `source`.
5. We establish `PART_OF` relationships between each `Chunk` node and its corresponding `Document` node.

Figure 5.2 illustrates the resulting graph structure after adding these relationships.

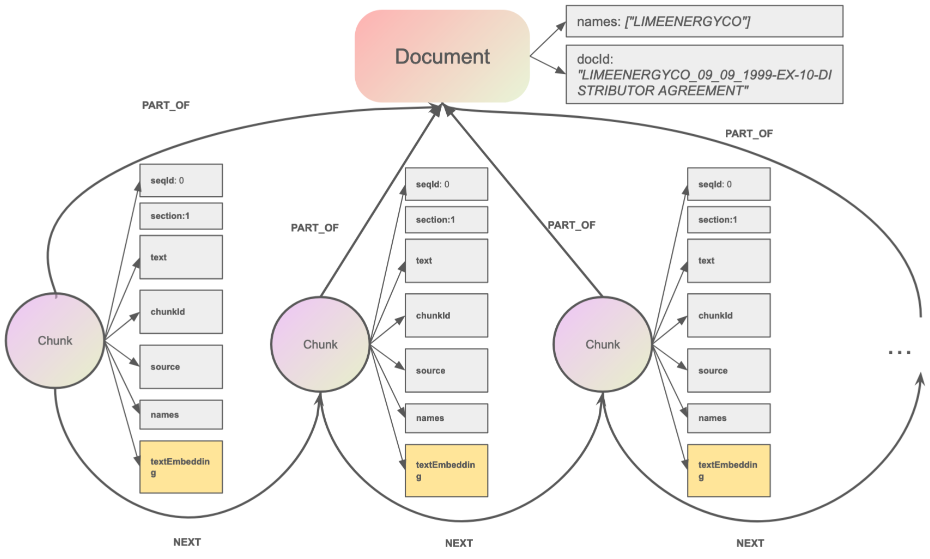


Figure 5.2: Graph structure with NEXT and PART_OF relationships

It's worth noting that these relationships do more than just maintain the structure of the original document. They enable contextual navigation, allowing our system to understand and utilize the document's inherent organization. For instance, when answering a query about a specific clause, our system can easily retrieve not just the clause itself, but also related clauses that come before or after it, providing a more comprehensive understanding.

5.1.3. Hierarchical Structure

The introduction of a hierarchical structure through Section nodes represents a significant enhancement to our knowledge graph. This structure mirrors the natural organization of most legal documents, which are typically divided into sections and subsections.

To further improve organization and facilitate more efficient querying, we introduce `Section` nodes as an intermediary layer between chunks and documents. Each `Section` node, as shown in Figure 5.3, contains:

1. `seqID`: The index of the section node in sequence
2. `sectID`: A unique identifier for the node, similar to `chunkId`
3. `names`: Names of the companies involved
4. `source`: Name of the source file
5. `summary`: A summary of the given section, generated by GPT-4
6. `summaryEmbedding`: Embedded vectors of the summary, generated using `text-embedding-3-small`

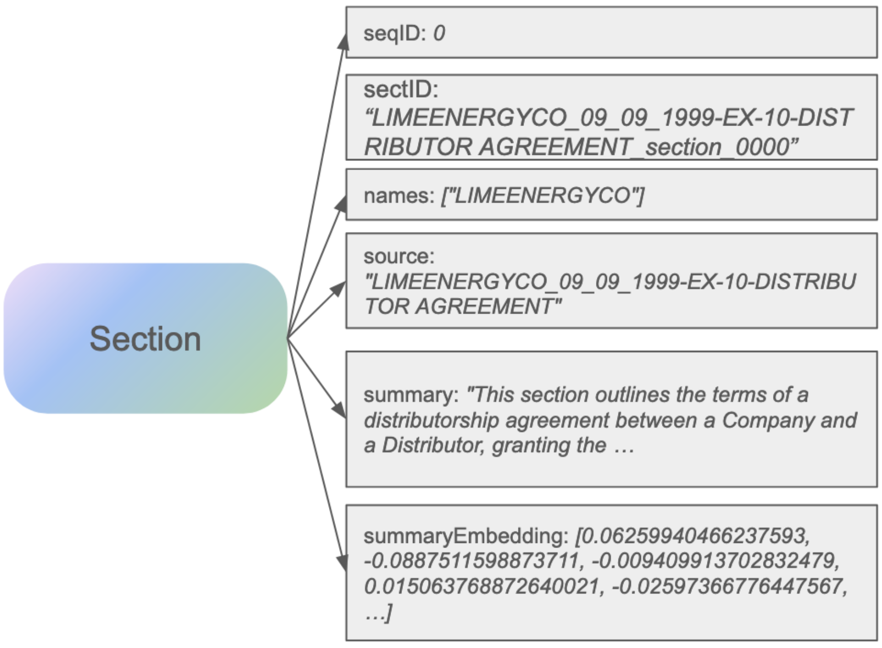

Section
seqID: 0
sectID: "LIMEENERGYCO_09_09_1999-EX-10-DISTRIBUTOR AGREEMENT_section_0000"
names: ["LIMEENERGYCO"]
source: "LIMEENERGYCO_09_09_1999-EX-10-DISTRIBUTOR AGREEMENT"
summary: "This section outlines the terms of a distributorship agreement between a Company and a Distributor, granting the …
summaryEmbedding: [0.06259940466237593, -0.0887511598873711, -0.009409913702832479, 0.015063768872640021, -0.02597366776447567, …]

Figure 5.3: Structure of a Section node in the knowledge graph

The process of creating and integrating `Section` nodes involves:

1. Identifying logical section breaks within documents (e.g., based on headings or content shifts).
2. Creating `Section` nodes with the properties listed above.
3. Generating summaries for each section using GPT-4.
4. Computing embeddings for these summaries using `text-embedding-3-small`.
5. Establishing `PART_OF` relationships from `Chunk` nodes to their respective `Section` nodes.
6. Creating `PART_OF` relationships from `Section` nodes to their parent `Document` nodes.

This results in a two-tier hierarchical knowledge graph, as illustrated in Figure 5.4.

The hierarchical structure offers several advantages:

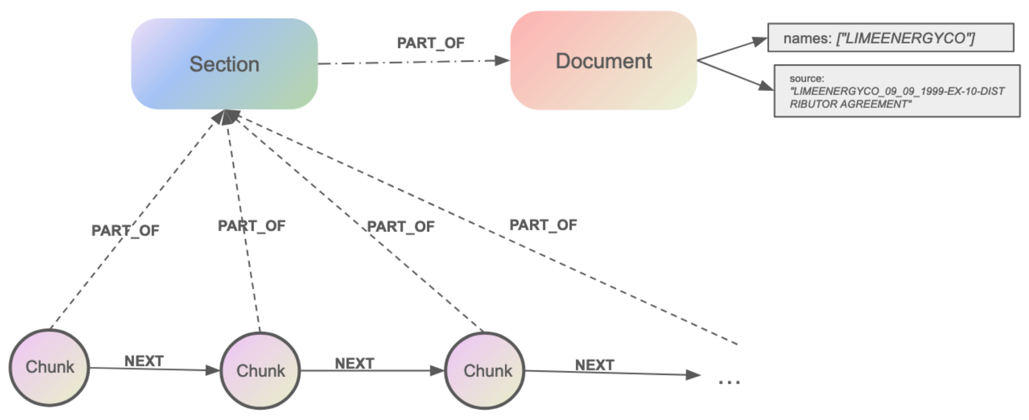


Figure 5.4: Two-tier hierarchical knowledge graph structure

- **Improved Navigation**: It allows for more efficient traversal of the document, mimicking how a human might navigate through a contract's table of contents.
- **Contextual Understanding**: By grouping related chunks under a Section node, we provide additional context that can be crucial for accurate interpretation of contract clauses.
- **Summarization at Multiple Levels**: The summaries generated for each Section node provide a higher-level overview of the document's content, which can be valuable for initial query routing or for providing concise answers to broad questions.
- **Flexibility in Querying**: This structure allows for queries at different levels of granularity - from broad questions about entire sections to specific queries about individual clauses.

5.2. Knowledge Graph Query Process

Our knowledge graph query process is designed to take full advantage of the rich structure and relationships encoded in our graph. It combines the efficiency of vector-based similarity search with the contextual awareness provided by our graph structure. The process follows these steps:

1. **Document Identification:** We first locate the relevant `Document` node based on the query context or user input.
2. **Section-Level Search:** We perform a similarity search on `Section` nodes connected to the identified document. This search uses the `summaryEmbedding` of each section and the embedding of the query to find the most relevant sections.
3. **Chunk-Level Search:** Within the identified relevant sections, we conduct another similarity search among the `Chunk` nodes. This fine-grained search helps pinpoint the most pertinent information.
4. **Context Retrieval:** We retrieve the most relevant `Chunk` node(s) based on the similarity search results. Additionally, we fetch the immediate neighbors (connected via `NEXT` relationships) to provide context.
5. **Response Generation:** The retrieved chunks, along with their sectional and document context, are then used as input for the LLM to generate a response to the original query.

This multi-level search approach offers several key benefits:

- **Efficiency**: By first identifying relevant sections before drilling down to specific chunks, we can quickly narrow our search space, reducing computational overhead.
- **Context Preservation**: By retrieving neighboring chunks along with the most relevant ones, we ensure that the LLM has access to the necessary context for accurate interpretation.
- **Flexibility**: This approach can handle both broad, high-level queries (which might be answered

sufficiently at the section level) and specific, detailed queries (which require chunk-level information).

- **Improved Accuracy**: By leveraging both the hierarchical structure and the semantic relationships encoded in our graph, we can provide more accurate and contextually relevant information to the LLM, potentially reducing hallucinations and improving the quality of generated responses.

In conclusion, our Knowledge Graph-based RAG system represents a significant advancement in the field of contract analysis. By encoding the structure and relationships inherent in legal documents, we create a system that can navigate and interpret these documents in a way that more closely mimics human understanding. This approach not only improves the accuracy and relevance of responses but also opens up new possibilities for contract analysis, such as identifying cross-references between different sections or documents, tracking definitions across a corpus of contracts, or even comparing the structure and content of multiple contracts.

As we move forward, there are several exciting avenues for further research and development. These include incorporating more sophisticated natural language processing techniques to automatically identify and classify different types of clauses, developing more advanced querying algorithms that can perform multi-hop reasoning across the graph, and exploring ways to visualize and interact with the knowledge graph to provide insights into contract structure and content.

5.3. The Case for Knowledge Graph-based RAG in Contract Analysis

As we delve deeper into the intricacies of contract analysis, it becomes increasingly apparent that traditional RAG systems, while powerful, often fall short in capturing the nuanced relationships inherent in legal documents. Our proposed Knowledge Graph-based RAG system emerges as a compelling solution to these challenges, offering a more sophisticated approach to understanding and navigating complex contractual landscapes.

5.3.1. Contextual Richness: Weaving the Fabric of Understanding

At the heart of our KG-based approach lies its ability to preserve and utilize the intricate web of relationships within contract documents. Unlike conventional RAG systems that view each text chunk in isolation, our method weaves these fragments into a coherent tapestry of information. This contextual richness is particularly crucial in the realm of legal analysis, where the interpretation of a single clause often hinges on its connections to various other elements scattered throughout the document.

Consider, for instance, the analysis of a complex liability clause. In a traditional RAG system, the retrieval process might surface this clause based on keyword similarity, but it could easily miss crucial context from related definitions or exceptions mentioned elsewhere in the contract. Our KG-based system, however, can effortlessly navigate these relationships, providing a holistic view that encompasses not just the clause itself, but also its broader contractual context.

This enhanced contextual understanding isn’t merely a theoretical advantage—it translates directly into more accurate and comprehensive contract interpretations. By enabling the LLM to ”see” the full picture, we reduce the risk of misinterpretations and increase the overall reliability of the analysis.

5.3.2. Navigating the Contractual Labyrinth

The hierarchical structure of our knowledge graph—comprising Document, Section, and Chunk nodes—mirrors the natural organization of legal documents. This isomorphism between the graph structure and document layout is more than just an elegant design choice; it fundamentally transforms how users can interact with and navigate through complex contracts.

Imagine a legal professional tasked with reviewing a lengthy merger agreement. With our system, they can start with a high-level overview of the document’s structure, easily identifying key sections of interest. From there, they can drill down into specific clauses, all while maintaining a clear sense of where each piece fits within the broader context of the agreement.

This multi-level navigation capability offers a stark contrast to the often disjointed experience of traditional keyword searches. It allows for a more intuitive exploration of documents, akin to the way an experienced lawyer might flip through a physical contract, quickly honing in on relevant sections while maintaining awareness of the document's overall structure.

5.3.3. From Fragments to Narratives: Enhancing Query Responses

Perhaps one of the most exciting aspects of our KG-based approach is its potential to transform the nature of query responses. Traditional RAG systems often provide answers that, while factually correct, can feel disjointed or lack narrative coherence. Our system, by contrast, has the potential to construct responses that tell a more complete story.

When queried about a specific contractual provision, our system doesn't just retrieve the relevant chunk of text. Instead, it can trace the provision's connections within the graph, identifying related clauses, relevant definitions, and even historical context from previous versions of the contract (if such information is encoded in the graph). This allows the LLM to generate responses that not only answer the immediate question but also provide valuable context and insights.

For example, a query about a change-of-control clause might yield a response that not only explains the clause itself but also highlights related provisions in the representations and warranties section, points out relevant definitions, and even draws attention to how this clause interacts with termination rights elsewhere in the contract.

5.3.4. Flexibility in the Face of Complexity

Legal documents are notoriously diverse and complex, ranging from straightforward nondisclosure agreements to intricate multi-party international contracts. Our KG-based RAG system offers the flexibility to handle this spectrum of complexity with grace.

For simpler documents, the system can provide quick, to-the-point answers based on direct node retrievals. For more complex analyses, it can leverage the full power of the graph structure, performing multi-hop reasoning to connect disparate pieces of information and provide nuanced, contextually rich responses.

This flexibility extends to the types of queries the system can handle. From broad questions about overall document structure to pinpoint inquiries about specific legal terms, our system adapts its response strategy based on the nature of the query and the structure of the relevant information in the graph.

5.4. Conclusion: Charting the Future of Contract Analysis

As we stand at the intersection of legal expertise and cutting-edge AI technology, our Knowledge Graph-based RAG system represents a step forward in the field of contract analysis. By encoding the structure and relationships inherent in legal documents, we've created a system that navigates and interprets these complex texts in a way that more closely mimics human understanding.

This approach not only promises more accurate and contextually relevant responses but also opens up new possibilities for contract analysis. From identifying cross-references between different sections or documents to tracking the evolution of clauses across multiple contract versions, the potential applications are vast and exciting.

As we look to the future, several avenues for further research and development beckon. These include the incorporation of more sophisticated natural language processing techniques to automatically identify and classify different types of clauses, the development of advanced querying algorithms capable of performing complex reasoning across the graph, and the exploration of interactive visualization techniques to provide intuitive insights into contract structure and content.

In conclusion, while traditional RAG systems have undoubtedly advanced the field of document analysis, our Knowledge Graph-based approach represents the next evolutionary step. By more faithfully representing the complex, interconnected nature of legal documents, we're not just improving contract analysis—we're reimagining it. As this technology continues to develop and mature, it holds the promise of transforming how legal professionals interact with and derive insights from contractual documents, ushering in a new era of efficiency and understanding in the legal domain.

6 Experiments

In this chapter, we present a comprehensive overview of our experimental setup designed to address our research questions. We will detail our choice of datasets, pipeline configurations, and evaluation metrics. This chapter is structured to provide a clear understanding of our methodological approach and the rationale behind our experimental design.

6.1. Research Questions and Hypotheses

To frame our experimental approach, we begin by revisiting our research questions, as initially presented in Section 1.2:

1. **How do RAG methodologies compare to long-context models in terms of effectiveness for legal document processing?**
2. **What is the impact of various enhancement techniques on the performance of RAG systems in contract review, and how do they affect both effectiveness and efficiency?**
3. **Can the integration of knowledge graphs, as an alternative to traditional vector databases, significantly improve RAG frameworks for legal information retrieval, given the structured nature of legal documents, such as contracts?**

For each research question, we have formulated specific hypotheses to guide our experimental design:

- **H1:** RAG methodologies will demonstrate comparable or superior effectiveness to long-context models in legal document processing, particularly for tasks requiring extensive contextual understanding.
- **H2:** Enhancement techniques, such as optimized chunking strategies and query expansion, will significantly improve both the effectiveness and efficiency of RAG systems in contract review tasks.
- **H3:** The integration of knowledge graphs into RAG frameworks will yield measurable improvements in retrieval accuracy and relevance for legal information retrieval tasks, compared to traditional vector database approaches.

6.2. Experimental Setup

Our experimental framework is designed to rigorously test our hypotheses and address our research questions. We employ two primary datasets: the Contract Understanding Atticus Dataset (CUAD) [27] and ContractNLI [34]. These datasets provide a robust foundation for evaluating our models in the context of legal document processing and contract review.

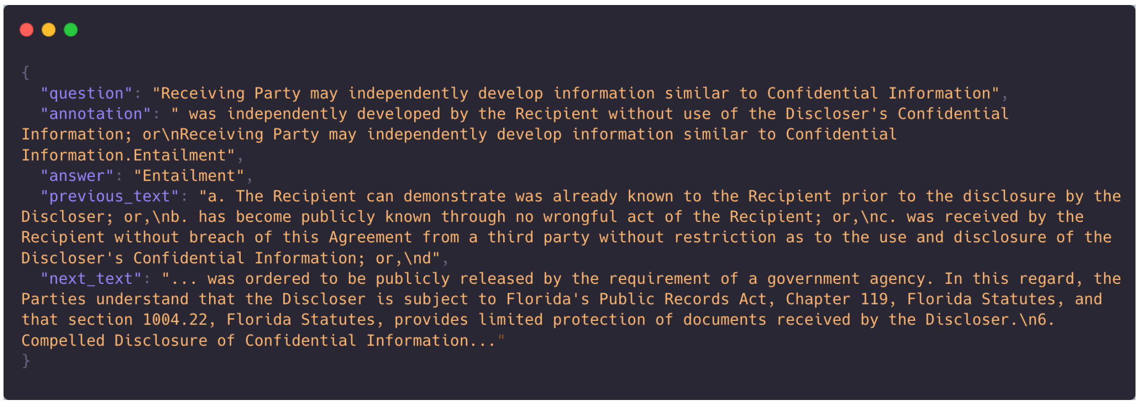
6.2.1. Datasets

ContractNLI Dataset

The ContractNLI dataset [34] serves as our primary benchmark for contract reviewing tasks. This dataset is specifically designed for document-level natural language inference on contracts, simulating the complex task of contract review. Each entry in the dataset consists of:

- A contract document
- A set of hypotheses (e.g., ”Some obligations of Agreement may survive termination.”)
- The task of determining whether each hypothesis is entailed by, contradicts, or is not mentioned (neutral) in the contract
- The requirement to identify evidence spans within the contract to support the decision

Figure 6.1 illustrates an example from the ContractNLI dataset:



```
{
  "question": "Receiving Party may independently develop information similar to Confidential Information",
  "annotation": " was independently developed by the Recipient without use of the Discloser's Confidential
Information; or\nReceiving Party may independently develop information similar to Confidential
Information.Entailment",
  "answer": "Entailment",
  "previous_text": "a. The Recipient can demonstrate was already known to the Recipient prior to the disclosure by the
Discloser; or,\nb. has become publicly known through no wrongful act of the Recipient; or,\nc. was received by the
Recipient without breach of this Agreement from a third party without restriction as to the use and disclosure of the
Discloser's Confidential Information; or,\nd",
  "next_text": "... was ordered to be publicly released by the requirement of a government agency. In this regard, the
Parties understand that the Discloser is subject to Florida's Public Records Act, Chapter 119, Florida Statutes, and
that section 1004.22, Florida Statutes, provides limited protection of documents received by the Discloser.\n6.
Compelled Disclosure of Confidential Information..."
}
```

Figure 6.1: Example of Contract NLI questions and evidence spans.

To gain deeper insights into the dataset’s characteristics, we conducted an analysis of the span lengths and label distributions, as shown in Figures 6.2 and 6.3:

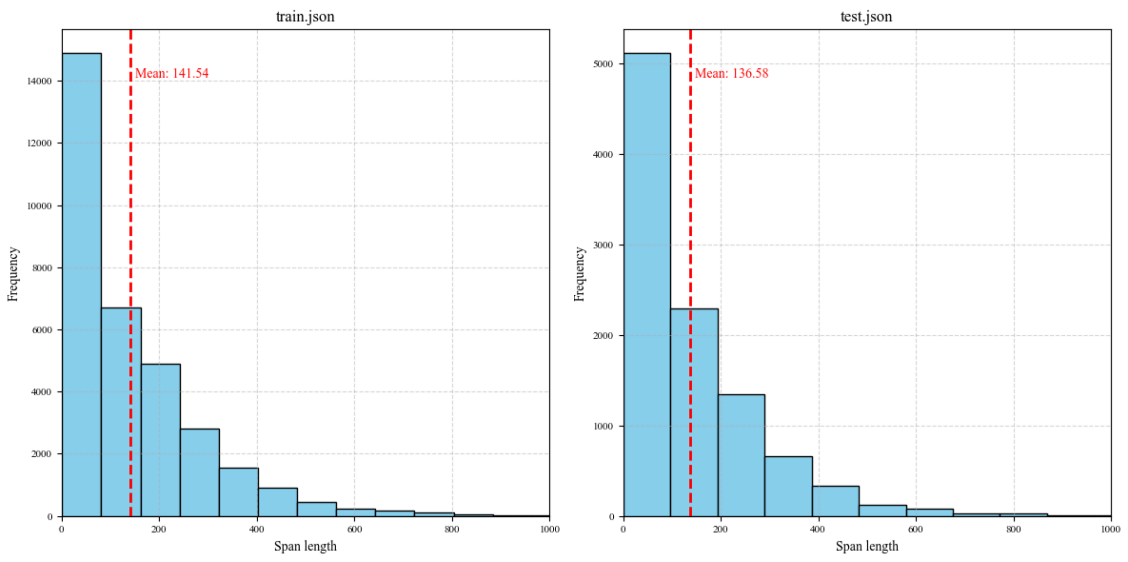


Figure 6.2: Distribution of evidence span lengths in the ContractNLI dataset.

Contract Understanding Atticus Dataset (CUAD)

The CUAD dataset [27] complements our experimental framework by providing a comprehensive collection of annotated contracts. Each entry in CUAD includes:

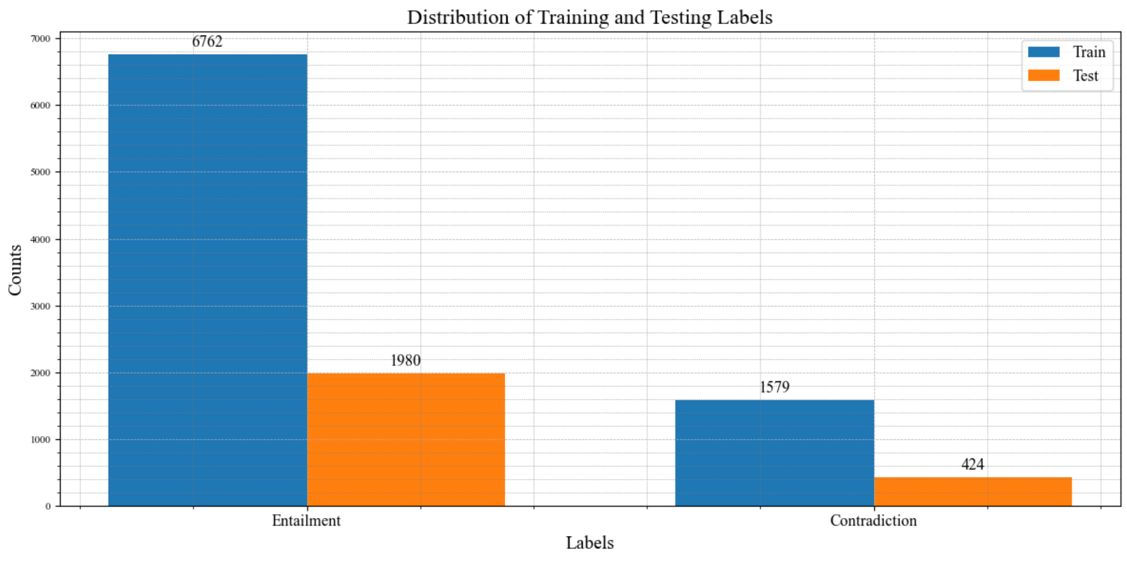


Figure 6.3: Label distribution in the ContractNLI dataset.

- A document identifier
- The document title
- A specific question about the contract
- The corresponding answers

An example entry from CUAD is structured as follows:

```
id: LIMEENERGYCO_09_09_1999-EX-10-DISTRIBUTOR AGREEMENT__Document Name
doc_title: LIMEENERGYCO_09_09_1999-EX-10-DISTRIBUTOR AGREEMENT
question: The name of the contract
answers: ['DISTRIBUTOR AGREEMENT']
q_with_title: In the document titled LIMEENERGYCO_09_09_1999-EX-10-DISTRIBUTOR
AGREEMENT, the following question was asked: The name of the contract
```

We utilize the `q_with_title` field as the query in our experiments with CUAD.

6.2.2. Experimental Design

Our experimental design is structured to systematically address each research question (RQ) through three distinct experiments.

Experiment 1: RAG vs. Long-Context Models (RQ1)

To evaluate the performance of RAG methodologies against long-context models, we developed two different experimental pipelines:

1. **RAG Pipeline:** Retrieves relevant document chunks from a vector database, based on similarity to the user question.
2. **Few-Shot Pipeline:** Leverages chain-of-thought reasoning, using the first five questions as in-context examples to guide the model's reasoning process.

Both pipelines were tested on 400 questions from the ContractNLI dataset, comparing results across commercial models such as Claude 3 and GPT-4. The specific model versions used include:

- Claude 3 Haiku: `claude-3-haiku-20240307`
- Claude 3 Sonnet: `claude-3-sonnet-20240229`
- Claude 3 Opus: `claude-3-opus-20240229`
- GPT-4: texttgpt-4-0613

Experiment 2: RAG Enhancement Techniques (RQ2)

To examine the impact of enhancement techniques on RAG performance, we focused on two main strategies:

1. **Chunking Strategies:**
    - Fixed size chunks (2048 characters)
    - Paragraph-based chunking
    - Recursive chunking with overlap (2048 characters, 200-character overlap)
2. **Query Expansion and Reranking:**
    - Query expansion (generating 3 alternative queries)
    - Reranking (selecting the top 3 results)

For query expansion, we used the following prompt to generate alternative user questions:

> *You are an AI assistant. Your task is to generate 3 different versions of the given user question to retrieve relevant documents from a vector database. By providing multiple perspectives, your goal is to overcome the limitations of similarity-based search. Provide the questions on separate lines.*

The evaluation dataset for this experiment consisted of 82 test cases from CUAD, derived from 41 contract labels, with two documents per label. GPT-4 was employed as the universal LLM for inferencing. The prompt for inferencing was as follows:

> *You are a lawyer reviewing a contract. When given a question, your task is to find the correct answer from the contract. Additional context is provided, but it may not always be relevant. Be honest and provide the accurate information. If you cannot determine the answer, say 'I don't know'. Keep your response concise.*

Incorporating "I don't know" as a valid response helps reduce the likelihood of incorrect answers, simplifying later evaluation and filtering of model outputs.

Experiment 3: Knowledge Graph Integration (RQ3)

In this experiment, we explored the integration of a knowledge graph to enhance retrieval performance. The setup was identical to Experiment 2, but with one key difference: instead of using a vector database, we replaced it with two versions of a knowledge graph (v1 with one layer of chunk nodes, and v2 with one layer of section nodes and one layer of chunk nodes connected to the first layer) for document retrieval. This allowed us to compare how knowledge graph-based systems impact the retrieval and reasoning process compared to traditional vector-based methods.

6.3. Evaluation Methodology

Our evaluation process comprises two stages: intermediate evaluation and final evaluation.

6.3.1. Intermediate Evaluation

For the intermediate stage, we assess:

- **Contextual Relevancy:** Using the DeepEval metric to measure the relevance of retrieved results to the query.
- **Ground Truth Presence:** Verifying if the ground truth is contained within the retrieved results.
- **Retrieval Efficiency:** Recording the time duration for each retrieval operation.

The contextual relevancy is calculated as:

$$\text{Contextual Relevancy} = \frac{\text{Number of Relevant Statements}}{\text{Total Number of Statements}} \tag{6.1}$$

6.3.2. Final Evaluation

For the final evaluation stage, we employ multiple metrics to comprehensively assess the performance of our models:

F1 Scores with Jaccard Similarity

To quantify the overlap between system output and ground truths, we use F1 scores calculated with Jaccard similarity. This approach, aligned with the evaluation method in the original CUAD paper [27], is particularly well-suited for contract review tasks where the exact wording may vary but the semantic content needs to match.

The F1 score is calculated as the harmonic mean of precision and recall:

$$F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{6.2}$$

Where precision and recall are defined as:

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \tag{6.3}$$

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{6.4}$$

In the context of our contract review tasks, we consider a prediction to be a true positive if its Jaccard similarity with the ground truth exceeds a certain threshold. The Jaccard similarity between two text spans A and B is calculated as:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{6.5}$$

Where $|A \cap B|$ represents the number of common words between A and B, and $|A \cup B|$ represents the total number of unique words in A and B combined.

We use a threshold of 0.5 for the Jaccard similarity, following the approach in the CUAD paper. This means that if the Jaccard similarity between a predicted span and the ground truth span is greater than 0.5, we consider it a correct prediction.

This metric is particularly useful for our tasks because:

- It accounts for partial matches, which are common in contract review where the exact wording might differ but the core content is the same.
- It balances precision (accuracy of the predictions made) and recall (coverage of all relevant information), which is crucial in legal document processing where both false positives and false negatives can have significant implications.
- It allows for a fair comparison between different models and techniques, including both RAG and long-context approaches.

The code for calculation is in Appendix A.1

6.4. Baseline Model

Our baseline consists of a standard RAG pipeline without additional optimizations. Key characteristics include:

- Default settings with K=3 and a similarity threshold of 0.6
- Embedding models and chat models sourced from OpenAI's suite of pre-trained models

This baseline serves as a point of comparison for our enhanced RAG techniques and knowledge graph integration experiments.

7 Results and Analysis

This chapter presents the findings from our experiments, offering a detailed analysis of the performance of various models and techniques in legal document processing tasks. We structure our results according to our three main research questions, providing insights into the comparative performance of RAG methodologies versus long-context models, the impact of enhancement techniques on RAG systems, and the efficacy of knowledge graph integration.

7.1. Experiment 1: Long-Context Models vs. RAG

Our first experiment aimed to compare the effectiveness and efficiency of RAG methodologies against long-context models in legal document processing. Table 7.1 summarizes the performance metrics and average inference times for different models on the Contract NLI task.

		Haiku	Sonnet	Opus	GPT-4
F1 Score	RAG	0.9585	0.9125	0.8865	0.951
	Few-shots	0.8895	0.9115	0.863	0.9325
Time Avg (s)	RAG	1.585 ± 0.970	3.366 ± 1.949	0.847 ± 0.453	1.612 ± 2.090
	Few-shots	2.318 ± 2.318	4.307 ± 2.608	1.109 ± 0.430	1.582 ± 1.528

Table 7.1: Performance Metrics and Average Inference Times for Different Models on Contract NLI

7.1.1. Performance Analysis

The results reveal interesting patterns across different models and approaches:

- **RAG Performance:** In the RAG pipeline, the Haiku model demonstrated superior performance with an F1 score of 0.9585, outperforming larger models like Sonnet (0.9125), Opus (0.8865), and even GPT-4 (0.951). This suggests that smaller, more specialized models can be highly effective when coupled with efficient retrieval mechanisms.
- **Few-Shot Learning:** When transitioning to the few-shot pipeline, which assesses a model's ability to adapt quickly with limited data, we observed a shift in performance rankings. GPT-4 emerged as the top performer with an F1 score of 0.9325, while other models experienced a decline in scores compared to their RAG counterparts. This indicates that larger models may have an advantage in few-shot scenarios, possibly due to their broader knowledge base.
- **Efficiency Considerations:** Regarding inference times, a crucial factor for real-world deployments, Haiku exhibited remarkable efficiency. It recorded the lowest average times for both RAG (1.585 ± 0.970 s) and few-shot (2.318 ± 2.318 s) pipelines. As expected, larger models like Sonnet and Opus incurred higher computational costs. Interestingly, despite its size, GPT-4 maintained competitive inference times, suggesting effective optimizations in its architecture.

These findings support our hypothesis H1, demonstrating that RAG methodologies can indeed offer comparable or superior effectiveness to long-context models in legal document processing, particularly for tasks requiring extensive contextual understanding.

7.1.2. Qualitative Analysis

To provide deeper insights into the models' performance, we conducted a qualitative analysis of their outputs. For instance, consider the following example response:

> *Based on the information provided, it's unclear whether the hypothesis "Confidential Information shall only include technical information" is entailed or contradicted. The text mentions that Confidential Information is being provided by the Broker regarding businesses for sale, but does not specify what types of information are included. Without more context about the scope of Confidential Information, there is not enough information to determine if the hypothesis is supported or refuted by the text.*

This response demonstrates the model's ability to:

- Accurately identify the lack of specific information in the given context.
- Refrain from making unfounded assumptions.
- Clearly articulate the reasoning behind its uncertainty.

Such nuanced responses are crucial in legal document processing, where misinterpretations can have significant consequences.

7.2. Experiment 2: Optimized RAG vs. Vanilla RAG

Our second experiment focused on evaluating the impact of various enhancement techniques on the performance of RAG systems in contract review tasks.

7.2.1. Chunking Optimization

Table 7.2 presents the results of our investigation into different chunking strategies.

Chunking Strategies	Contextual Relevance	F1 Score
fixed_size_2048	**0.247**	0.426
paragraph	0.228	0.5
recursive_2048	**0.247**	**0.501**

Table 7.2: Comparison of Different Chunking Techniques

Key findings include:

- The `recursive_2048` strategy emerged as the most effective approach, achieving the highest F1 score (0.501) while maintaining the same contextual relevance (0.247) as the baseline `fixed_size_2048` strategy.
- The paragraph-based chunking strategy, despite yielding a lower contextual relevance score, showed a marked improvement in F1 score compared to the baseline.
- These results suggest that the choice of chunking strategy can significantly impact the performance of RAG systems, with the recursive approach offering a balanced improvement in both relevance and accuracy.

7.2.2. Pre-retrieval and Post-retrieval Optimization

Table 7.3 presents the results of our investigation into advanced retrieval techniques.

Contrary to expectations, our analysis revealed:

- The baseline approach outperformed both the reranking and the combined query expansion with reranking methods across all measured metrics.

Pipelines	Contextual Relevance	F1 Score	Avg Time (s)
Baseline	**0.247**	**0.501**	**1.433**
Reranking	0.183	0.388	2.643
Query Expansion + Reranking	0.224	0.374	3.732

Table 7.3: Comparison of Retrieval Optimization Techniques

- Advanced techniques demonstrated inferior performance, with the reranking method yielding the lowest contextual relevance (0.183) and a reduced F1 score (0.388).
- The combination of query expansion and reranking showed only marginal improvement in contextual relevance (0.224) but the lowest F1 score (0.374).
- Notably, these sophisticated approaches significantly increased processing times, with the combined method requiring more than twice the time of the baseline (3.732 seconds).

These results challenge the assumption that more complex retrieval techniques inherently lead to improved performance, highlighting the need for careful consideration of the trade-offs between retrieval quality and computational efficiency in practical applications.

7.3. Experiment 3: Knowledge Graph + RAG vs. Vector DB + RAG

Our final experiment aimed to evaluate the efficacy of integrating knowledge graphs into RAG frameworks compared to traditional vector database approaches. Table 7.4 summarizes our findings.

Strategies	Contextual Relevance	F1 Score	Avg Time (s)
Vector Database	0.247	0.501	**1.433**
Knowledge Graph V1	**0.287**	0.484	1.753
Knowledge Graph V2	0.274	**0.521**	1.864

Table 7.4: Comparison between Knowledge Graph and Vector Database Approaches

Key observations include:

- The vector database approach demonstrates a balanced performance profile, achieving an F1 score of 0.501 and a contextual relevance of 0.247, while maintaining the lowest average processing time of 1.433 seconds.
- Knowledge Graph V1 shows the highest contextual relevance (0.287) among all strategies, indicating enhanced semantic understanding, though its F1 score (0.484) is slightly lower than the vector database.
- Knowledge Graph V2 achieves the best F1 score (0.521) and maintains a high contextual relevance (0.274), suggesting a more balanced approach to retrieval quality.
- Both knowledge graph implementations incur a time penalty, with V1 and V2 requiring 1.753 and 1.864 seconds on average, respectively.

These results partially support our hypothesis H3, demonstrating that the integration of knowledge graphs into RAG frameworks can yield measurable improvements in retrieval accuracy and relevance for legal information retrieval tasks, albeit at the cost of increased processing time.

7.4. Summary of Findings

Our experiments have yielded several important insights:

1. RAG methodologies can outperform long-context models in certain legal document processing tasks, particularly when using smaller, specialized models like Haiku.
2. The choice of chunking strategy significantly impacts RAG performance, with recursive approaches offering a good balance between relevance and accuracy.

3. Contrary to expectations, more complex retrieval techniques did not consistently improve performance, emphasizing the importance of balancing sophistication with efficiency.
4. Knowledge graph integration in RAG frameworks shows promise in enhancing semantic understanding and retrieval quality, but at the cost of increased processing time.

These findings have important implications for the design and implementation of AI systems for legal document processing, highlighting the need for careful optimization and trade-off considerations in real-world applications.

7.5. Implications and Future Research Directions

The findings from our experiments have several important implications for both research and practice in legal document processing:

7.5.1. Theoretical Implications

Our results challenge several conventional assumptions in the field:

- **Model Size and Performance:** The superior performance of smaller models like Haiku in RAG implementations suggests that model size may not be the determining factor in legal document processing tasks. This finding calls for a reevaluation of the ”bigger is better” paradigm in language model development.
- **Complexity vs. Efficiency:** The unexpected underperformance of sophisticated retrieval techniques indicates that increased complexity does not necessarily yield better results. This suggests the need for a more nuanced understanding of how different components in RAG systems interact.
- **Knowledge Representation:** The promising results from knowledge graph integration, despite the performance overhead, point to the importance of structured knowledge representation in legal document understanding.

7.5.2. Practical Implications

For practitioners and system designers, our findings suggest several key considerations:

- **System Design:** When implementing legal document processing systems, organizations should carefully consider the trade-off between model size, processing speed, and accuracy. Our results suggest that smaller, specialized models with efficient RAG implementations may offer the best balance for many applications.
- **Resource Allocation:** The minimal performance gains from complex retrieval techniques suggest that resources might be better allocated to optimizing basic RAG components rather than implementing sophisticated enhancements.
- **Knowledge Integration:** While knowledge graphs show promise, their implementation should be carefully evaluated against the specific needs of the application, given the processing overhead they introduce.

7.5.3. Future Research Directions

Our research findings point toward several exciting opportunities for future investigation. We envision four main areas where further research could significantly advance the field of legal document processing:

1. **Hybrid Architectures:** Perhaps the most promising direction lies in developing systems that intelligently combine multiple approaches. Future research should explore mechanisms that seamlessly transition between RAG and long-context models based on the specific needs of each legal task. Additionally, we see great potential in building bridges between vector databases and knowledge graphs to leverage their complementary strengths while minimizing their individual weaknesses. This work could focus particularly on designing streamlined knowledge graph implementations that maintain rich context while reducing the computational overhead that currently limits their practical application.

2. **Performance Optimization:** Our experiments revealed several opportunities for enhancing system efficiency. Future work should focus on developing smarter chunking strategies that can adapt to the unique structure of different legal documents, whether they're contracts, briefs, or regulatory texts. This could be complemented by creating more efficient pathways through legal knowledge graphs, potentially drawing inspiration from how legal professionals navigate complex document relationships. We also see potential in finding innovative ways to compress and streamline sophisticated retrieval methods without sacrificing their effectiveness, particularly in time-sensitive legal applications.
3. **Legal Domain Specialization:** Given our surprising findings about model size and performance, we see rich potential in further specialization of AI systems for legal tasks. This includes adapting existing AI models to better understand legal nuances through refined transfer learning techniques. Equally important is the creation of new evaluation metrics that better reflect real-world requirements of legal document processing. Of particular interest is the exploration of specialized training approaches for smaller models, which our research suggests might be more efficient and effective for specific legal tasks than their larger counterparts.
4. **Scale and Performance:** As legal document collections grow increasingly vast, understanding scale becomes crucial. Future research needs to examine how our systems perform across different document sizes and complexity levels, from simple contracts to intricate legal frameworks. This investigation should include ways to distribute processing across systems efficiently, especially for large-scale document analysis. Additionally, developing smart caching strategies could significantly speed up access to frequently referenced legal information, potentially transforming how we handle repeated queries in legal document processing.

These directions build directly on our experimental findings while addressing key challenges we encountered. The surprising effectiveness of simpler models and approaches suggests that innovative, targeted solutions might be more valuable than simply scaling up existing methods. By pursuing these research directions, we believe the field can make significant strides toward more effective, efficient, and practical legal document processing systems.

7.5.4. Limitations and Methodological Considerations

Future research should address several limitations of the current study:

- **Dataset Diversity:** Expansion of experiments to include a broader range of legal document types and jurisdictions
- **Long-term Performance:** Investigation of system performance over extended periods and varying workloads
- **Real-world Validation:** Conducting studies in actual legal practice settings to validate laboratory findings
- **Cost-benefit Analysis:** Development of comprehensive frameworks for evaluating the economic implications of different system architectures

These research directions would significantly advance our understanding of effective legal document processing systems and their practical implementation.

8 Conclusion

This research advances the field of legal document processing through innovative combinations of Retrieval-Augmented Generation (RAG) and knowledge graph (KG) integration. Our work demonstrates the potential of combining artificial intelligence approaches to address complex challenges in legal document analysis and processing, specifically contract review.

Our experiments revealed that for long-context legal text understanding and analysis, RAG architectures consistently outperform in-context learning approaches. Perhaps most intriguingly, we discovered that the smaller Haiku model achieved remarkably high performance in RAG implementations, challenging conventional assumptions about model size and effectiveness in legal applications.

Through controlled experiments, we systematically investigated techniques to enhance the base RAG system. The recursive chunking strategy emerged as particularly effective for document processing, while surprisingly, we found that simpler retrieval approaches often matched or exceeded the performance of their more complex counterparts. This suggests that while techniques like reranking and query expansion can benefit legal document processing, their application should be carefully considered.

To address the limitations of vector databases, we developed a novel approach using knowledge graphs to represent contracts, providing richer and interconnected context. Our method maintains chunk-level granularity while establishing meaningful connections between segments, organized in a hierarchical structure by sections. The retrieval process leverages this hierarchy, conducting similarity searches from section-level to chunk-level granularity when applicable, and returns the most relevant nodes along with their conceptually related neighbors.

This integrated approach demonstrated superior performance compared to traditional RAG systems using vector databases, achieving both higher contextual relevance and improved accuracy in final outputs.

8.1. Summary of Contributions

The primary contribution of this research lies in the novel integration of knowledge graphs into the RAG framework, significantly enhancing contextual understanding and retrieval accuracy in legal document processing. By enabling systems to comprehend complex relationships within legal documents, this integrated approach marks a substantial advancement toward more interpretable AI systems in the legal domain.

Our research has also yielded several additional key findings. We demonstrated that RAG methodologies utilizing smaller, specialized models can outperform larger, general models in specific legal tasks. This finding challenges the conventional wisdom that larger models invariably deliver better results, suggesting more efficient, task-specific solutions for legal document processing.

Furthermore, our investigation of chunking strategies for RAG revealed that recursive chunking techniques provide an optimal balance between relevance and accuracy. This practical approach offers immediate benefits for document processing workflows in legal environments.

The research also addresses the critical balance between performance and computational efficiency in legal AI applications. Given the time-sensitive nature of legal practice, our findings provide actionable insights for optimizing AI system deployment in real-world legal settings.

8.2. Limitations

While our study presents significant advances, several limitations warrant acknowledgment:

- Dataset Scope: Our experiments, while robust, were limited to the ContractNLI and CUAD datasets, representing only a subset of legal document types. The diverse nature of legal documents across different systems and domains presents ongoing challenges for generalization.
- Rapid Evolution of AI Models: Our research focused on contemporary models (Claude 3.0 family and GPT-4). The fast-paced evolution of AI technology means newer models may offer different advantages or superior performance.
- Resource Constraints: Available computational resources limited our experimental scope. While comprehensive, larger-scale studies might reveal additional performance nuances.
- External Knowledge Integration: While our KG-RAG integration showed promise, opportunities remain for incorporating broader external legal knowledge and transforming it into structured formats for enhanced understanding.
- Lack of Practitioner Evaluation: Though technically thorough, our analysis would benefit from legal practitioner feedback to better align AI outputs with practical requirements.

8.3. Future Research Directions

Our findings open several promising avenues for future research in legal document processing. Perhaps the most pressing challenge lies in optimizing knowledge graph integration for practical deployment. While our current approach demonstrates significant improvements in accuracy, there remains substantial room for reducing computational overhead. We envision developing intelligent systems that can dynamically adjust their use of knowledge graph components based on query complexity, potentially achieving the best of both worlds: the rich context of graph-based approaches with the speed of simpler retrieval methods.

The surprising effectiveness of the Haiku model in our experiments suggests an exciting direction for model development. Rather than following the trend toward ever-larger models, future research might focus on creating specialized, efficient models specifically trained for legal document processing. This could include developing new pre-training approaches that leverage legal domain knowledge and creating adaptive processing mechanisms that build upon our successful recursive chunking strategy.

Our findings on retrieval architectures also merit deeper investigation. The observation that simpler approaches often matched complex ones raises intriguing questions about when and how to deploy advanced retrieval techniques. Future work should explore developing intelligent systems that can adapt their retrieval strategy based on the specific characteristics of legal documents and queries, potentially leading to more efficient and effective processing pipelines.

Finally, we believe that bridging the gap between research and practice should be a key priority. This includes conducting systematic evaluations in real-world legal settings, particularly focusing on our most promising configurations like the Haiku-RAG system with recursive chunking. Such real-world validation would not only verify our findings but also provide crucial insights into practical implementation challenges and opportunities for improvement.

8.4. Summary

This research advances the understanding of AI applications in legal document processing, particularly through the innovative integration of RAG systems and knowledge graphs. While significant progress has been made, the field presents ongoing challenges and opportunities. The legal domain's demands for accuracy, interpretability, and ethical considerations necessitate continued research to refine AI models for these complex requirements. Realizing the full potential of AI in law will require sustained collaboration between technologists, legal professionals, and ethicists to ensure AI systems effectively and equitably serve the pursuit of justice.

References

[1] Abdelrahman Abdallah, Bhawna Piryani and Adam Jatowt. ‘Exploring the State of the Art in Legal QA Systems’. In: *Journal of Big Data* 10.1 (Aug. 2023), p. 127. ISSN: 2196-1115. DOI: 10.1186/s40537-023-00802-8. arXiv: 2304.06623 [cs]. (Visited on 17/11/2023).

[2] Mohammed Al Qady and Amr Kandil. ‘Concept relation extraction from construction documents using natural language processing’. In: *Journal of construction engineering and management* 136.3 (2010), pp. 294–302.

[3] Ebtesam Almazrouei et al. *The Falcon Series of Open Language Models*. Nov. 2023. DOI: 10.48550/arXiv.2311.16867. arXiv: 2311.16867 [cs]. (Visited on 11/03/2024).

[4] ARCADIS. *2022 Global Construction Disputes Report*. 2022. URL: https://www.arcadis.com/en-gb/knowledge-hub/perspectives/global/global-construction-disputes-report.

[5] Deniz Artan Ilter and Gokce Bakioglu. ‘Modeling the relationship between risk and dispute in subcontractor contracts’. In: *Journal of Legal Affairs and Dispute Resolution in Engineering and Construction* 10.1 (2018), p. 04517022.

[6] Sören Auer et al. ‘Dbpedia: A nucleus for a web of open data’. In: *international semantic web conference*. Springer. 2007, pp. 722–735.

[7] Jinze Bai et al. *Qwen Technical Report*. Sept. 2023. DOI: 10.48550/arXiv.2309.16609. arXiv: 2309.16609 [cs]. (Visited on 11/03/2024).

[8] Yuntao Bai et al. ‘Constitutional ai: Harmlessness from ai feedback’. In: *arXiv preprint arXiv:2212.08073* (2022).

[9] Emily M Bender et al. ‘On the dangers of stochastic parrots: Can language models be too big?□□’. In: *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*. 2021, pp. 610–623.

[10] Luiz Bonifacio et al. *InPars: Data Augmentation for Information Retrieval Using Large Language Models*. Feb. 2022. DOI: 10.48550/arXiv.2202.05144. arXiv: 2202.05144 [cs]. (Visited on 15/01/2024).

[11] Tom Brown et al. ‘Language Models Are Few-Shot Learners’. In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. (Visited on 06/12/2023).

[12] Ali Bedii Candaş and Onur Behzat Tokdemir. ‘Automating coordination efforts for reviewing construction contracts with multilabel text classification’. In: *Journal of Construction Engineering and Management* 148.6 (2022), p. 04022027.

[13] So-Won Choi and Eul-Bum Lee. ‘Contractor’s risk analysis of engineering procurement and construction (EPC) contracts using ontological semantic model and Bi-long short-term memory (LSTM) technology’. In: *Sustainability* 14.11 (2022), p. 6938.

[14] Pierre Colombo et al. *SaulLM-7B: A Pioneering Large Language Model for Law*. Mar. 2024. DOI: 10.48550/arXiv.2403.03883. arXiv: 2403.03883 [cs]. (Visited on 12/03/2024).

[15] Jiaxi Cui et al. *ChatLaw: Open-Source Legal Large Language Model with Integrated External Knowledge Bases*. June 2023. DOI: 10.48550/arXiv.2306.16092. arXiv: 2306.16092 [cs]. (Visited on 17/01/2024).

[16] Prithiviraj Damodaran. *FlashRank, Lightest and Fastest 2nd Stage Reranker for search pipelines.* Version 1.0.0. Dec. 2023. DOI: 10.5281/zenodo.10426927. URL: https://github.com/PrithivirajDamodaran/FlashRank.

[17] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. May 2019. DOI: 10.48550/arXiv.1810.04805. arXiv: 1810.04805 [cs]. (Visited on 16/11/2023).

[18] Jialin Dong et al. ‘Don’t Forget to Connect! Improving RAG with Graph-based Reranking’. In: *arXiv preprint arXiv:2405.18414* (2024). (Visited on 24/06/2024).

[19] Weili Fang et al. ‘Automated text classification of near-misses from safety reports: An improved deep learning approach’. In: *Advanced Engineering Informatics* 44 (2020), p. 101060.

[20] Dieter Fensel et al. ‘Introduction: what is a knowledge graph?’ In: *Knowledge graphs: Methodology, tools and selected use cases* (2020), pp. 1–10.

[21] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. Jan. 2024. DOI: 10.48550/arXiv.2312.10997. arXiv: 2312.10997 [cs]. (Visited on 07/02/2024).

[22] Randy Goebel et al. ‘Summary of the competition on legal information, extraction/entailment (COLIEE) 2023’. In: *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*. 2023, pp. 472–480.

[23] Neel Guha et al. ‘Legalbench: Prototyping a collaborative benchmark for legal reasoning’. In: *arXiv preprint arXiv:2209.06120* (2022).

[24] Kelvin Guu et al. *REALM: Retrieval-Augmented Language Model Pre-Training*. Feb. 2020. DOI: 10.48550/arXiv.2002.08909. arXiv: 2002.08909 [cs]. (Visited on 06/02/2024).

[25] Fahad ul Hassan, Tuyen Le and Duc-Hoc Tran. ‘Multi-class categorization of design-build contract requirements using text mining and natural language processing techniques’. In: *Construction Research Congress 2020*. American Society of Civil Engineers Reston, VA. 2020, pp. 1266–1274.

[26] Fahad ul Hassan and Tuyen Le. ‘Automated requirements identification from construction contract documents using natural language processing’. In: *Journal of Legal Affairs and Dispute Resolution in Engineering and Construction* 12.2 (2020), p. 04520009.

[27] Dan Hendrycks et al. *CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review*. Nov. 2021. DOI: 10.48550/arXiv.2103.06268. arXiv: 2103.06268 [cs]. (Visited on 11/12/2023).

[28] Quzhe Huang et al. ‘Lawyer llama technical report’. In: *arXiv preprint arXiv:2305.15062* (2023).

[29] Gautier Izacard and Edouard Grave. *Distilling Knowledge from Reader to Retriever for Question Answering*. Aug. 2022. DOI: 10.48550/arXiv.2012.04584. arXiv: 2012.04584 [cs]. (Visited on 01/02/2024).

[30] Albert Q. Jiang et al. *Mistral 7B*. Oct. 2023. DOI: 10.48550/arXiv.2310.06825. arXiv: 2310.06825 [cs]. (Visited on 11/03/2024).

[31] Bowen Jin et al. ‘Large language models on graphs: A comprehensive survey’. In: *arXiv preprint arXiv:2312.02783* (2023).

[32] Daniel Martin Katz et al. ‘Gpt-4 passes the bar exam’. In: *Philosophical Transactions of the Royal Society A* 382.2270 (2024), p. 20230254.

[33] Youyi Kim et al. ‘Application of natural language processing (NLP) and text-mining of big-data to engineering-procurement-construction (EPC) bid and contract documents’. In: *2020 6th conference on data science and machine learning applications (CDMA)*. IEEE. 2020, pp. 123–128.

[34] Yuta Koreeda and Christopher Manning. ‘ContractNLI: A Dataset for Document-level Natural Language Inference for Contracts’. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Ed. by Marie-Francine Moens et al. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 1907–1919. DOI: 10.18653/v1/2021.findings-emnlp.164. (Visited on 11/12/2023).

[35] LangChain. *RAG from Scratch*. Accessed: 2024-06-11. 2024. URL: https://github.com/langchain-ai/rag-from-scratch.

[36] Jeehee Lee, June-Seong Yi and JeongWook Son. ‘Development of automatic-extraction model of poisonous clauses in international construction contracts using rule-based NLP’. In: *Journal of Computing in Civil Engineering* 33.3 (2019), p. 04019003.

[37] JeeHee Lee et al. ‘Effective risk positioning through automated identification of missing contract conditions from the contractor’s perspective based on FIDIC contract cases’. In: *Journal of Management in Engineering* 36.3 (2020), p. 05020003.

[38] Patrick Lewis et al. ‘Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks’. In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474. (Visited on 15/01/2024).

[39] Huayang Li et al. *A Survey on Retrieval-Augmented Text Generation*. Feb. 2022. DOI: 10.48550/arXiv.2202.01110. arXiv: 2202.01110 [cs]. (Visited on 06/02/2024).

[40] Han Liu, Alexander Gegov and Frederic Stahl. ‘Categorization and construction of rule based systems’. In: *Engineering Applications of Neural Networks: 15th International Conference, EANN 2014, Sofia, Bulgaria, September 5-7, 2014. Proceedings 15*. Springer. 2014, pp. 183–194.

[41] Linhao Luo et al. ‘Reasoning on graphs: Faithful and interpretable large language model reasoning’. In: *arXiv preprint arXiv:2310.01061* (2023).

[42] Gary Marcus. ‘The next decade in AI: four steps towards robust artificial intelligence’. In: *arXiv preprint arXiv:2002.06177* (2020).

[43] Lauren Martin et al. *Better Call GPT, Comparing Large Language Models Against Lawyers*. Jan. 2024. DOI: 10.48550/arXiv.2401.16212. arXiv: 2401.16212 [cs]. (Visited on 11/03/2024).

[44] Ariana Martino, Michael Iannelli and Coleen Truong. ‘Knowledge injection to counter large language model (LLM) hallucination’. In: *European Semantic Web Conference*. Springer. 2023, pp. 182–185.

[45] Joshua Maynez et al. ‘On faithfulness and factuality in abstractive summarization’. In: *arXiv preprint arXiv:2005.00661* (2020).

[46] Seonghyeon Moon, Seokho Chi and Seok-Been Im. ‘Automated detection of contractual risk clauses from construction specifications using bidirectional encoder representations from transformers (BERT)’. In: *Automation in Construction* 142 (2022), p. 104465.

[47] Seonghyeon Moon et al. ‘Automated construction specification review with named entity recognition using natural language processing’. In: *Journal of Construction Engineering and Management* 147.1 (2021), p. 04020147.

[48] Niklas Muennighoff et al. ‘MTEB: Massive text embedding benchmark’. In: *arXiv preprint arXiv:2210.07316* (2022). (Visited on 20/06/2024).

[49] Humza Naveed et al. *A Comprehensive Overview of Large Language Models*. Nov. 2023. DOI: 10.48550/arXiv.2307.06435. arXiv: 2307.06435 [cs]. (Visited on 20/11/2023).

[50] OpenAI et al. *GPT-4 Technical Report*. Dec. 2023. DOI: 10.48550/arXiv.2303.08774. arXiv: 2303.08774 [cs]. (Visited on 07/02/2024).

[51] David Patterson et al. ‘Carbon emissions and large neural network training’. In: *arXiv preprint arXiv:2104.10350* (2021).

[52] Fabio Petroni et al. ‘Language models as knowledge bases?’ In: *arXiv preprint arXiv:1909.01066* (2019).

[53] Hung Phan et al. ‘RAG vs. Long Context: Examining Frontier Large Language Models for Environmental Review Document Comprehension’. In: *arXiv preprint arXiv:2407.07321* (2024).

[54] Zhixiao Qi et al. ‘Foodgpt: A large language model in food testing domain with incremental pre-training and knowledge graph prompt’. In: *arXiv preprint arXiv:2308.10173* (2023).

[55] Alec Radford and Karthik Narasimhan. ‘Improving Language Understanding by Generative Pre-Training’. In: 2018. (Visited on 07/02/2024).

[56] Adam Roberts, Colin Raffel and Noam Shazeer. *How Much Knowledge Can You Pack Into the Parameters of a Language Model?* Oct. 2020. DOI: 10.48550/arXiv.2002.08910. arXiv: 2002.08910 [cs, stat]. (Visited on 01/02/2024).

[57] Guilherme Moraes Rosa et al. ‘To Tune or Not to Tune? Zero-Shot Models for Legal Case Entailment’. In: *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*. ICAIL ’21. New York, NY, USA: Association for Computing Machinery, July 2021, pp. 295–300. ISBN: 978-1-4503-8526-8. DOI: 10.1145/3462757.3466103. (Visited on 13/11/2023).

[58] Spurthi Setty et al. *Improving Retrieval for RAG Based Question Answering Models on Financial Documents*. Mar. 2024. DOI: 10.48550/arXiv.2404.07221. arXiv: 2404.07221 [cs, q-fin]. (Visited on 11/06/2024).

[59] Fabian M Suchanek, Gjergji Kasneci and Gerhard Weikum. 'Yago: a core of semantic knowledge'. In: *Proceedings of the 16th international conference on World Wide Web*. 2007, pp. 697–706.

[60] Jiashuo Sun et al. 'Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph'. In: *arXiv preprint arXiv:2307.07697* (2023).

[61] Yi Tay et al. 'Scaling laws vs model architectures: How does inductive bias influence scaling?' In: *arXiv preprint arXiv:2207.10551* (2022).

[62] Llama team. 'The Llama 3 Herd of Models'. In: *arXiv* (July 2024). HUMAN & MACHINE INTELLIGENCE CONVERSATIONAL AI. URL: https://ai.meta.com/research/publications/the-llama-3-herd-of-models/ (visited on 31/07/2024).

[63] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. July 2023. DOI: 10.48550/arXiv.2307.09288. arXiv: 2307.09288 [cs]. (Visited on 06/12/2023).

[64] Ashish Vaswani et al. *Attention Is All You Need*. Aug. 2023. DOI: 10.48550/arXiv.1706.03762. arXiv: 1706.03762 [cs]. (Visited on 07/02/2024).

[65] Denny Vrandečić and Markus Krötzsch. 'Wikidata: a free collaborative knowledgebase'. In: *Communications of the ACM* 57.10 (2014), pp. 78–85.

[66] Jason Wei et al. 'Chain-of-thought prompting elicits reasoning in large language models'. In: *Advances in neural information processing systems* 35 (2022), pp. 24824–24837.

[67] Yilin Wen, Zifeng Wang and Jimeng Sun. 'Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models'. In: *arXiv preprint arXiv:2308.09729* (2023).

[68] Saika Wong et al. 'Construction Contract Risk Identification Based on Knowledge-Augmented Language Models'. In: *Computers in Industry* 157–158 (May 2024), p. 104082. ISSN: 0166-3615. DOI: 10.1016/j.compind.2024.104082. (Visited on 11/06/2024).

[69] X Xue, Y Hou and J Zhang. 'Automated construction contract summarization using natural language processing and deep learning'. In: *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*. Vol. 39. IAARC Publications. 2022, pp. 459–466.

[70] Jianxiong Yang et al. 'Machine learning–driven model to analyze particular conditions of contracts: A multifunctional and risk perspective'. In: *Journal of Management in Engineering* 38.5 (2022), p. 04022036.

[71] Linyao Yang et al. 'Chatgpt is not enough: Enhancing large language models with knowledge graphs for fact-aware language modeling'. In: *arXiv preprint arXiv:2306.11489* (2023).

[72] Antonio Jimeno Yepes et al. 'Financial Report Chunking for Effective Retrieval Augmented Generation'. In: *arXiv preprint arXiv:2402.05131* (2024).

[73] Jiansong Zhang and Nora M El-Gohary. 'Extending building information models semiautomatically using semantic natural language processing techniques'. In: *Journal of Computing in Civil Engineering* 30.5 (2016), p. C4016004.

A appendix

A.1. CUAD Evaluation

```python
import pandas as pd
import numpy as np
from sklearn import metrics

IOU_THRESH = 0.3

def get_jaccard(gt, pred):
    remove_tokens = [".", ",", ";", ":"]
    for token in remove_tokens:
        gt = gt.replace(token, "")
        pred = pred.replace(token, "")
    gt = gt.lower()
    pred = pred.lower()
    gt = gt.replace("/", "␣")
    pred = pred.replace("/", "␣")

    gt_words = set(gt.split("␣"))
    pred_words = set(pred.split("␣"))

    intersection = gt_words.intersection(pred_words)
    union = gt_words.union(pred_words)
    jaccard = len(intersection) / len(union)
    return jaccard

def compute_precision_recall(df, predictions):
    tp, fp, fn = 0, 0, 0

    for i, row in df.iterrows():
        answers = eval(row['answers'])  # Assuming 'answers' is stored as a string representation of a list
        pred = predictions[i]

        if len(answers) == 0:
            if pred:
                fp += 1
        else:
            match_found = False
            for ans in answers:
                if get_jaccard(ans, pred) >= IOU_THRESH:
                    match_found = True
                    break

            if match_found:
                tp += 1
            else:
                fn += 1

        if not match_found and pred:
            fp += 1

    precision = tp / (tp + fp) if tp + fp > 0 else np.nan
    recall = tp / (tp + fn) if tp + fn > 0 else np.nan

    return precision, recall
```