A REPORT

ON

LEGALRAG: AN LLM BASED RAG SYSTEM TO PROVIDE LEGAL LITERACY AND SUPPORT

BY

Name of the Student: Sachin Shankar Hebbar

ID.No. : 2022AC05016

AT

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

Pilani (Rajasthan) INDIA

LegalRAG: an LLM based RAG system to provide legal literacy and support

DISSERTATION

Submitted in partial fulfillment of the requirements of the

Degree : MTech in Artificial Intelligence and Machine Learning.

By

Sachin Shankar Hebbar
2022AC05016

Under the supervision of

Naveen Rathani, Quantitative Analytics Manager

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
Pilani (Rajasthan) INDIA

March, 2025

Acknowledgements

I would like to express my sincere gratitude to everyone who supported me throughout the dissertation.

First and foremost, I extend my heartfelt appreciation to my supervisor, Naveen Rathani, for his invaluable guidance, encouragement, and insights throughout the semester. His expertise and continuous support have played a crucial role in shaping this project.

I am immensely thankful to my evaluator, Prof. Seetha Parameswaran, for her valuable feedback and constructive suggestions. Insights provided by Prof. Seetha have helped me refine my work and enhance the quality of this project.

I would also like to thank BITS Pilani for providing this amazing opportunity to pursue M.Tech while working. Project work has benefited a lot from all the many of the subjects I learned in the past two years at BITS Pilani.

Lastly, I extend my gratitude to my family, friends, classmates, and colleagues who have supported me in various ways during this journey.

Thank you all for your invaluable contributions.

Sachin Shankar Hebbar
BITS Pilani Work Integrated Learning Programme (WILP)
M.Tech in AI and ML
2022AC05016

Certificate from the supervisor (attach screenshot)

Abstract

The complex world of India's legal system, with its long history and vast literature, can intimidate many non-professionals. However, everyone needs to have a reasonable level of legal literacy to protect themselves from harm of different kinds. The advent of Large Language Models[1] (LLM), in conjunction with the availability of vast amounts of public data regarding laws, acts, and cases, provides an opportunity to build a system that can act as the bridge between the legal system and the common people.

This work aims to build LegalRAG, a RAG-based system that uses an LLM to provide relevant responses and resources to users' legal queries related to Legal acts of inheritance, divorce, copyright, and consumer protection.

LLMs without any fine-tuning can be used to accomplish the task mentioned above. LLMs might have included legal information in their training datasets; they also carry tremendous knowledge through their parameters. However, this is general knowledge and is not explicitly tuned to contain Indian legal understanding. Other methods, like vanilla fine-tuning and Low-Rank Adapters (LoRA)[2], would fine-tune the LLMs and equip them to answer legal queries, but these require more memory and resources to fine-tune the additional layers.

The proposed method of LegalRAG includes intent recognition followed by Retrieval Augmented Generation[3] (RAG) architecture. User query passes through an intent recognition model to identify user intent: inheritance, divorce, consumer protection, or copyright-related intent. Depending on the intent, the user query is routed to different RAG systems or knowledge bases.

RAG is a method that combines the LLM's generation capabilities with information retrieval capabilities. Each of the four RAG systems takes in the user query, embeds it, and searches a vector store or knowledge base of Indian Legal data corresponding to one of the four acts. Most relevant context from the search and the user query are sent to the LLM to generate a response.

The hindrances from the previous approaches are resolved through RAG-based architecture as they provide the domain knowledge the LLM would need to generate relevant responses and not hallucinate. This approach requires fine-tuning the intent recognition model but doesn't mandatorily require any fine-tuning of the LLM itself. Therefore, it consumes less time and resources. RAG has the additional benefit of source

verification by providing information on the sources, such as the Act name and section number containing the selected context.

The novelty in this idea is with the intent recognition model, as more legal acts get included in the future; only the intent recognition model, which is much smaller in size than the LLM, needs to be re-trained without many updates to the LLM, therefore this approach requires less memory and resources..

LegalRAG would also include a Named Entity Recognition[4] model to identify Personally Identifiable Information of any individuals and mask them before storing the data in vector stores or knowledge bases to protect the individual's data.

LegalRAG will benefit the general public by providing essential legal literacy and offering procedural steps and templates for filing cases under the abovementioned acts.

Key Words:

Large Language Model (LLM), Named Entity Recognition (NER), Intent Recognition, fine-tuning, Low-Rank Adapters, Retrieval Augmented Generation (RAG), Embedding model, Natural Language Processing (NLP), Legal Literacy

List of Symbols and Abbreviations used

Abbreviation	Full Form
RAG	Retrieval Augmented Generation
PII	Personally Identifiable Information
LLM	Large Language Model
LoRA	Low Rank Adapter
Q-LoRA	Quantized Low Rank Adapter
BAAI	Beijing Academy of Artificial Intelligence
BGE	BAAI General Embedding
HyDE	Hypothetical Document Embeddings
NER	Named Entity Recognition
GPT	Generative Pre-trained Transformer
TF-IDF	Term Frequency - Inverse Document Frequency
BM - 25	Best Matching - 25
LSA	Latent Semantic Analysis
CRF	Conditional Random Fields
SVM	Support Vector Machine
BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Networks
PAN	Permanent Account Number
CPU	Central Processing Unit
IVF	Inverted File
PPO	Proximal Policy Optimization

DPO	Direct Preference Optimization
BPE	Byte-Pair Encoding

List of Tables

List of Figures

Table of Contents

Chapter 1 - Introduction

The Indian Legal system has a vast history. Early parts of it started during the rule of the British in India. Over time, it has gone through several changes. Numerous acts and revisions have been made to it. Going through all the acts of the Indian legal system and gaining a good understanding can be overwhelming. It is difficult to understand the nooks and corners of the legal system with a busy life and there's no easy roadmap to get started on it. Due to this, most of the common folks of India are unaware of the Indian Legal system. However, having a basic understanding of the Indian legal system can be very beneficial to many as they find themselves in unfortunate situations.

Project 'LegalRAG' is a bridge between the vast world of the Indian legal system and the common folks of India. It sorts through the information and provides the most relevant legal response that people are looking for.

Purpose of LegalRAG:

LegalRAG aims at building a system that would make it easier for the general public to access legal literacy and basic legal support using the latest advancements in the field of LLMs (Large Language Models).

For the purpose of this project, legal acts and data related to inheritance, divorce, consumer protection and copyright laws in India will be used. No other legal acts and areas will be used.

LegalRAG's process involves an intent classification model that routes the user's textual query to the relevant act. The routed query is used to search through the relevant act's vector store and relevant context to answer the query is retrieved. The retrieved context and the user query are sent to the LLM to get the response that answers the user query.

The methodology and architecture is described in detail in the chapter 'Methodology'.

Expected Outcome of LegalRAG:

LegalRAG would take in textual user query related to legal systems of India and output a relevant textual response along with any additional resources and sources cited to ensure the response is trustworthy.

Objectives:

The following are the objectives listed in the abstract submission document of the project LegalRAG:

- To provide legal literacy to the general public of India.
- To provide basic legal support for filing cases to the common folks of India.
- To provide source verification and resources for all the responses the system provides and enhance user trust.
- To ensure consumer data is protected and masked.

The **first objective** regarding legal literacy is **completely met** as the project provides relevant legal responses related to the four acts of inheritance, divorce, consumer protection and copyright, therefore it paves way for legal literacy to the general public.

The **second objective** of providing basic legal support for filing cases and procedures has been **completely met** as the vector stores contain information on filing cases and procedures related to the four acts mentioned earlier.

The **third objective** of source verification and resources have been **completely met** as LegalRAG produces the top 5 retrieved context for every response that it provides to user queries. The context would contain the document name from which the context has been extracted.

The **fourth objective** is related to identifying and protecting consumer's PII (Personally Identifiable Information) through masking before storing them in the vector stores has been **completely met**. This objective was met through a Named Entity Recognition model that identifies all PII data and replaces them with a mask, such as ****. This masking ensures no person's PII data gets revealed to the users of LegalRAG inadvertently while providing a response.

Some of the keywords used here will be explained in the later chapters in more detail.

Chapter 2 - Literature Review

This chapter provides details on the literature survey performed before choosing the Methodology that is described in the next chapter.

It should be noted that currently, there is no existing process that does what LegalRAG intends to do for supporting legal literacy and support to the common public of India. However, with the rise of Large Language Models, RAG systems and Generative AI many similar processes exist that try to achieve similar things in other fields. The Literature survey performed goes through the different methodologies that exist today that could perform the activities LegalRAG does. The next chapter explains why the current methodology was chosen.

Many open source (Llama[5], Zephyr[6], Deepseek[7]) and proprietary (openAI[8], Claude[9], Mistral[10) Large Language models (LLM) exist today that can answer questions related to the Indian legal system without any fine-tuning. This could be because Indian legal data may have been part of the training process of these LLMs. However, LLMs may not have knowledge of all aspects of the Indian legal system. Additionally, Indian legal data is not static. It changes over time, new cases and new rulings keep getting added which makes it difficult for the LLM to capture as they were not part of the training.

It could also be because of the generalization capabilities of the LLMs where they can use what they learned about different country's legal systems and use that to answer Indian legal queries. However, this may not always lead to accurate responses.

Some of the latest proprietary LLMs (like GPT 4[11] or GPT o1[12]) can scrape the internet and get the results regarding Indian legal data even if it was not part of the training. However, this feature has a cost as openAI and other proprietary LLM research companies only provide a few free credits for use.

Since LLMs as it is can not always provide an accurate response on Indian legal data, there is a need for fine-tuning. There are many ways of fine-tuning an LLM. There are methods that don't involve parameter updates such as prompt tuning. In prompt tuning, you provide instructions to the LLM along with some examples (few shot learning[13]) on how you want the response to be. Prompt tuning[14] would not be suitable for answering Indian legal queries because it is more suitable to teach LLM to do a particular task rather than seeking knowledge.

There are also few-shot learning methods or prefix tuning[15] methods that can be done while fine-tuning the LLM or during the training itself. These methods involve parameter updates but as mentioned earlier few-shot learning is useful for teaching LLM to learn specific types of tasks like summarizing, translation etc. and not for seeking knowledge on a particular topic like the Indian legal system.

There are other methods for fine-tuning an LLM so it gains the capability or understanding to answer Indian legal queries. Adapters[16] are one of them. Adapters is a light weight fine-tuning method, where few new layers called adapters are added to the LLM. Freeze all the layers of the LLM except the adapters and use Indian legal data queries and responses to fine-tune the LLM. Only the adapter layer parameters will be updated. This fine-tuning method can improve the capability of the LLM to accurately respond to the Indian legal system. It's less resource intensive because parameter update only happens to the adapter layers. However, it increases the size of the LLM and even though it improves the performance of the LLM there are other techniques that have better performance.

LoRA[2] (Low Rank Adapters) is a technique that has shown better performance than Adapters. LoRA is a fine-tuning technique that is similar to Adapter, as it adds new layers to the LLM. It works similar to adapter fine-tuning where only the LoRA layers get updated during the fine-tuning and all other parameters are frozen. LoRA uses a linear algebra approximation where matrices of lower ranks and dimensions can be used to approximate a matrix of a higher rank and dimension. Therefore, compared to adapters, LoRA adds a lesser number of new parameters. So it reduces the compute resources lower than adapters during fine-tuning. It also requires less memory than adapters.

Q-LoRA[17] is Quantized LoRA, it is exactly the same as LoRA but the parameter values of the adapters are quantized to lower precision so they consume less space than LoRA technique. It may reduce the performance slightly, as the parameter values become approximate after quantization but it's a trade-off for saving space and giving up some accuracy.

Although adapters, LoRA and Q-LoRA are fine-tuning methods that can improve LLM’s capability to accurately respond to Indian legal queries. These are relatively resource intensive as they add new layers and require fine-tuning which also takes time. Another issue with these fine-tuning methods is that every time legal data changes or as new things get added, the fine-tuning process will have to be repeated. So, it's not cost effective.

There is a need for a different technique that doesn’t require fine-tuning the LLM every time there is an update in the Indian legal system. This is where Retrieval Augmented Generation (RAG) is useful. RAG combines information retrieval methods and the LLM language generation capabilities. The chapter on RAG provides more information on RAG. It also explains some of the retrieval techniques used in LegalRAG. The retrieval techniques used in LegalRAG are based on cosine similarity, and neural re-rankers which have shown better performance than traditional retrieval methods like using linear algebra techniques like TF-IDF[18], and probabilistic methods like Latent Semantic Analysis[19] (LSA) and Best Matching -25[20] (BM-25).

Some of the advanced retrieval techniques considered for LegalRAG are query decomposition[21], query fusion[22], Hypothetical Document Embeddings[23] (HyDE) and the chapter on results show the performance through each of these methods.

Justification for the choice of RAG:

1. It doesn’t mandatorily require fine-tuning of the LLM

2. The knowledge base of Indian legal data can update at any frequency and it doesn’t require any updates to the LLM to ensure LegalRAG’s responses are still accurate and up-to-date. The only update happens to the retrieval process which is much easier and faster.

LegalRAG also uses an intent classification model built fine-tuning distilBERT[24]. The chapter on intent classification explains that in further detail. There are many ways to perform intent classification from simple rule based or keywords based methods, or using machine learning methods like Naive Bayes[25] or Decision Trees[26]. However, these methods don’t have the level of language understanding that transformer models like distilBERT does. Therefore distilBERT was chosen as the model for intent classification.

It should be noted that even an LLM with few-shot learning or instruction tuning can be used for intent classification. It may even lead to a better intent classification model than the one with distilBERT. However, it was not done as LLM takes much more space and resources than distilBERT. Additionally as it can be seen in the Results that distilBERT is performing very well, showing there is no need to add compute and memory intensive LLM for a marginal improvement in performance.

LegalRAG also includes a Named Entity Recognition (NER) model. There are rule-based and keyword based methods to identify named entities. These methods don’t always generalize well and it's very difficult to create an extensive list of rules and keywords. There are machine learning based methods like Conditional Random Fields[27] (CRF), decision trees and Support Vector Machine[28] (SVM) based methods. These methods perform better than rule-based methods but they don’t have natural language understanding unlike transformer models like BERT[29] and distilBERT. Transformer based NER models have better performance than the previous methods.

There are also CNN based NER models provided by spaCy[30] which is what is used in LegalRAG as it is much smaller in size than transformer models and still has a very good performance based on previous research.

Several evaluation metrics have been reviewed to test LegalRAG in its intent classification capability, context relevance[31], faithfulness[32] of the response and other classification metrics. These are listed in the chapter on Evaluation Metrics.

Chapter 3 - Methodology

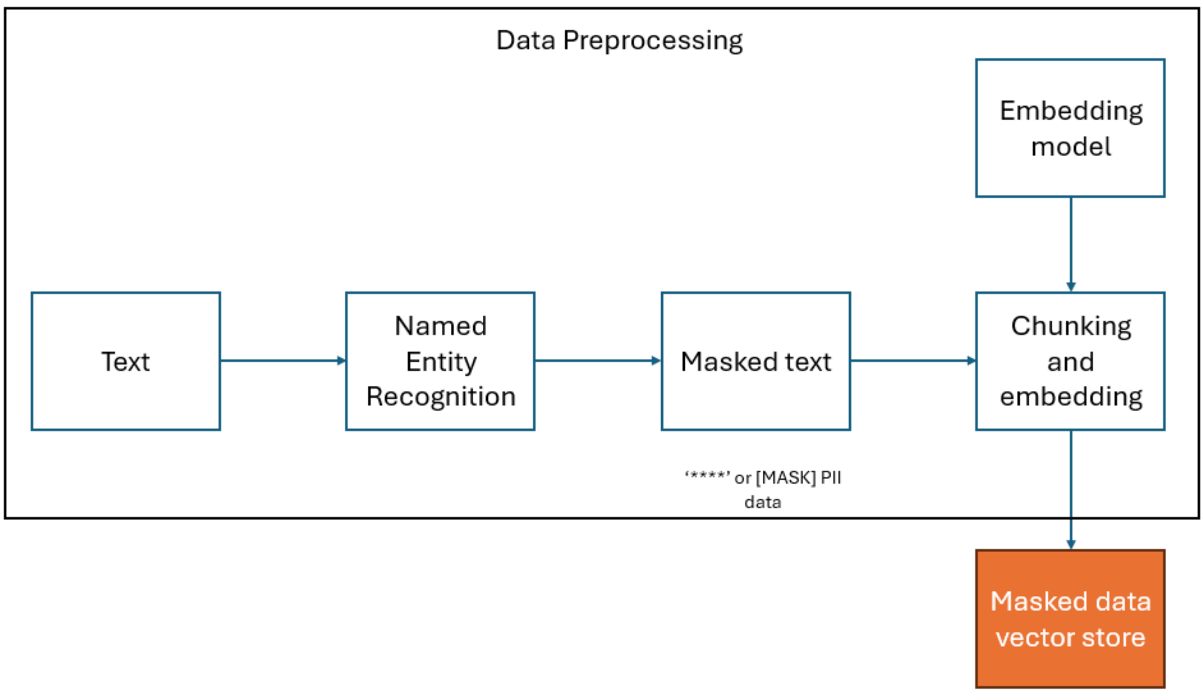
To provide relevant legal response to user query, LegalRAG has three main components:

1. Data pre-processing/Named Entity Recognition to mask the PII data
2. Intent Classification to route the user query to relevant vector store
3. RAG system for each of the four acts

Each of the above components have sub-components attached to it and there are many interactions between these components.

The below two architectures show how these three components that make up LegalRAG interact with each other.

Figure 1: Data pre-processing architecture



The above architecture shows the data pre-processing section of the architecture. The textual data from each of the four acts (inheritance, copyright, consumer protection, or divorce) will be passed through a Named Entity Recognition (NER) model. The NER model identifies all the relevant PII data. The list of all the PII data identified is listed in the Chapter on Named Entity Recognition. The Identified PII is then masked, in other words, the PII data is replaced with '****' to protect them.

The masked data goes through the steps of Chunking and embedding. Chunking and Embedding are part of the RAG system. Chunking divides the masked text into smaller pieces of text (or sentences) and each of these chunks are embedded into a numerical vector in the embedding step. Embedding is performed using an Embedding model.

More information on Chunking, embedding and embedding model can be seen in the chapter on RAG systems.

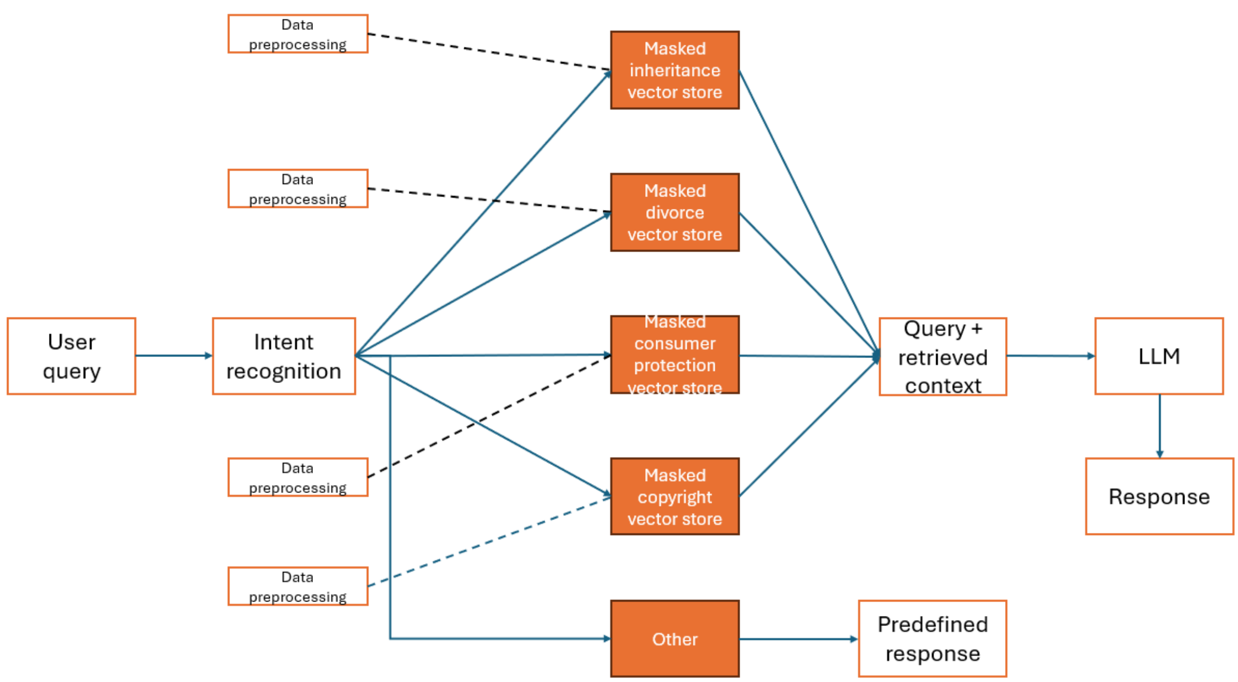
After Chunking and embedding, the numerical vectors are stored in a vector store. The user query will search through this vector store to get the relevant context for the response.

More information on the vector store is available in the chapter on RAG systems.

The above data pre-processing steps happen for all the four acts that were listed earlier.

The second architecture shows the overall architecture of the LegalRAG. The data pre-processing step shown in the previous figure and how it fits in the overall scene can be seen in the below figure.

Figure 2: LegalRAG architecture including data preprocessing



In the above architecture, whenever a user sends a textual query to the LegalRAG. There is an Intent classification model that identifies the intent of the user query. The intent can be one of the following: Inheritance, Divorce, Consumer protection, Copyright, or Other. Based on the intent of the query, the query is routed to the relevant vector store.

If the intent is Other, that means the user query is unrelated to the four acts of the legal system that is of interest. These 'Other' intent query will be given a predefined response asking the user query to ask queries related to the four acts.

Data pre-processing steps that happened before saving the data in vector store format was shown in the previous figure.

After the query is routed to the relevant vector store, it goes through advanced retrieval techniques to search through the vector store and get the most relevant context from the vector store. These advanced retrieval techniques are part of the RAG system. The retrieval techniques considered are the simple retrieval using cosine similarity, query fusion, and Hypothetical Document Embeddings (HyDE).

All these retrieval techniques are explained further in the chapter on retrieval techniques.

After the relevant context is retrieved, the user query and the context is passed to the LLM which generates the response that answers the user query. The RAG system also ensures the relevant context, and the document names from which the context is retrieved are provided. This information provides source verification and builds trust with the user on the LLM response.

Further chapters explain each of the components in the architecture in detail and then discuss the evaluation of the entire LegalRAG system, with results and conclusion.

Novelty in the approach:

The addition of intent classification is the novelty in the approach. Generally, the models used for intent classification will be much smaller than the LLM in size and memory. Fine-tuning the intent classification model requires much less resources and much less time than fine-tuning the LLM.

The advantage is, more legal acts (other than the four acts mentioned before) can be added into new vector stores and no updates or fine-tuning needs to be done to the LLM. Only the intent classification model will have to be fine-tuned on new intents so that it will consider queries related to the new legal acts. The entire architecture remains the same, except a data preprocessing step, a new vector store related to the new legal act gets added and the intent classification model will have the capacity to classify queries into new classes of intent. No change happens to the LLM.

This is very advantageous when LegalRAG is deployed in an application. New updates of the LegalRAG would only need an update to the intent classification model while the rest of the architecture remains the same.

Another advantage brought by the intent classification model is that this structure can be used for other similar problems too. For example, instead of answering legal act related information, it can be used to solve questions from different fields of science like classical physics, astronomy, quantum mechanics etc. Even in this case, the vector store would contain data related to science, and no updates will be made to the LLM. Only the intent classification model will be fine-tuned to route the user query to the relevant vector store containing different scientific information. Some changes to the chunking and retrieval techniques may be needed as the problem type changes, these changes may increase the storage slightly, but they don’t require additional computational resources.

Chapter 4 - Data Preprocessing

Data preprocessing is the first step or component of LegalRAG (see Figure 1). This component has multiple sub-components in it. This chapter will be a deep-dive into each of the sub-components and why they are needed.

Data preprocessing is performed on data related to all four acts of inheritance, copyright, consumer protection and divorce (as can be seen in Figure 2). Data preprocessing has the following sub-components:

1. Text with PII data
2. Named Entity Recognition (NER) model
3. Masking the PII data in the text
4. Chunking
5. Embedding using an embedding model
6. Indexing and storing in a vector store

Text with PII data:

The data collected related to the four legal acts can contain PII data of people involved in court cases for consumer protection, divorce, inheritance or copyright. These PII data need to be protected. To protect these data, there has to be a way to identify the PII data automatically instead of manually looking for them. The NER model is the automated way that can help identify these PII data.

Following are the PII data that can appear in each of the four acts and the NER model can identify them:

PII data present in inheritance act:

- Names of the deceased person and heirs
- Date of birth and age of individuals
- Family relationship and names (example: spouse, children, siblings)
- Aadhaar number, PAN
- contact numbers
- valuation of the inheritance
- address of the inheritance location and will information

PII data present in divorce act:

- Names of both spouses
- date of birth and age of individuals
- date and place of marriage, marriage certificate information
- names and ages of children, dependents, guardians involved
- residential address and phone number
- aadhaar number and PAN

PII data present in consumer protection act:
- Name of the complainant (consumer) and the respondent (business)
- Age and date of birth
- address and phone number, email addresses
- credit card or debit card information
- dates of transaction

PII data present in copyright act:
- Name of the copyright owner
- date of birth and age
- aadhaar, PAN and phone number
- Copyrighted work case and docket numbers
- Contracts, agreements and license numbers

Named Entity Recognition (NER) model:

A named entity is a word or phrase that identifies a specific person, place, thing or idea. An NER model is used to identify the different types of named entities available in the text.

spaCy, a free python package, provides an NER model called en_core_web_sm that is used to identify all the PII data in the text, in the context of LegalRAG.

Reason for choosing spaCy's NER model en_core_web_sm[33]:
1. It is open-source
2. It consumes less memory and runs efficiently on CPU (low computational resources)
3. Several applications and research papers have used spaCy
4. spaCy provides good bug support and has a good community to answer questions.
5. En_core_web_sm is a smaller, lightweight model and specialized on English language which makes it a suitable model as the text is in English
6. En_core_web_sm has very good performance on benchmark datasets as well as on the text used for LegalRAG

The following are the types of named entities that en_core_web_sm can identify:
1. Person's name (PERSON)
2. Organization's name (ORG)
3. Geo-political entities (GPE)
4. Date and time (DATE)
5. Monetary values (MONEY)
6. Nationalities, Religion and Political groups (NORP)

Below steps show how this NER model was trained:
1. The model was trained on OneNote 5 corpus[34] that contains data from news articles and web.
2. Word embeddings and part of speech tagging and dependency relations were extracted and used as features
3. A CNN model was trained that would identify the named entities.

Below steps show how NER model identifies Named entities or PII data:

1. Splits the text into tokens
2. Converts each of the word or token in the text into a word embedding
3. Extracts Part of Speech tags and dependency relations from the tokens
4. Passes the work embeddings and extracted information to the CNN model and gets output on the named entities.
5. Perform post processing to resolve any ambiguous dependency.

Masking the PII text:

This is a simple step which takes in all the named entity PII data extracted from the spaCy's NER model and replace them with '****' in the text. This masking step ensures no PII data of the person involved in court cases will be inadvertently shared with the users of LegalRAG as it is trying to provide the relevant response to the user query.

Chunking:

Chunking is a part of RAG, but in the case of LegalRAG it is more suitable to keep it as a part of the Data preprocessing step.

Chunking involves splitting the data into multiple chunks where each chunk can be a group of sentences or a group of characters. In the case of LegalRAG, each chunk will be a group of sentences.

Chunking is required as it helps localize the context and helps the data get stored in the vector store in a more meaningful way. It also makes searching the vector store much easier as specific context will be part of smaller chunks rather than long documents.

There are many ways of performing this chunking, but in the case of LegalRAG the following method was considered.
Chunk type: Sentence Splitter[35]
Chunk size: 100 (means each chunk will contain 100 sentences)
Chunk overlap: 50 (each chunk would contain 50 sentences from the previous chunk)

Chunk overlap is required as it provides additional context and forms a continuation with the previous chunk and the next chunk. This makes it easier to store the data in the vector store and also in the retrieval of the chunks. When in the RAG step, LLM generates a response, the LLM has more context in it due to the chunk overlap.

Reason for the choice of the above explained method for chunking over other methods:

1. Chunking size and overlap was chosen experimentally based on the document size and its performance on the evaluation dataset and user query response.
2. This is the simplest yet efficient way of chunking the data.

An example of sentence splitting that shows the structure of sentence splitting making into chunks, For the sake of this example, chunk size is considered to be 2, and chunk overlap is 1.

Here is the full document:

Sentence 1 and Sentence 2 and Sentence 3 and Sentence 4 and Sentence 5 and Sentence 6

After sentence chunking, this is how each of the chunks would look like:
Chunk 1: Sentence 1 and Sentence 2
Chunk 2: Sentence 2 and Sentence 3
Chunk 3: Sentence 3 and Sentence 4
Chunk 4: Sentence 4 and Sentence 5
Chunk 5: Sentence 5 and Sentence 6

Embedding:

Embedding the data into numerical vectors is a part of RAG, but in the case of LegalRAG it is more suitable to keep it as a part of the Data preprocessing step.
After chunking the data, each chunk has to be converted to a numerical vector embedding. This can be done through an embedding model. It should be noted that each chunk will be converted to an embedding and not each word within the chunk that gets converted to a vector embedding.

The embedding model used here is bge-small-en-v1.5[36] (BAAI general Embedding small). This embedding model produces embeddings of length 1024.

Reason for choosing this embedding model:
1. It is open-source
2. 1024 is relatively a smaller embedding size which makes the memory consumed by it lesser.
3. Bge-small-en-v1.5 is a smaller embedding model and therefore lesser space when compared to the embedding models provided by openAI (which also have a cost associated with it).
4. Bge-small-en-v1.5 has a good track record of being used in many RAG applications.

Indexing and storing in Vector store:

Indexing and vector store storage are part of RAG, but in the case of LegalRAG it is more suitable to keep it as a part of the Data preprocessing step.

Indexing data in a vector store is like savings books in different sections in a library. Similar to how easy it becomes to retrieve a book based on its section number, it becomes easier and faster to retrieve data that is indexed.

In the context of vector store, indexing refers to the way the similar numerical vector embeddings are stored together and dissimilar vector embeddings are saved apart. Various indexing techniques exist to store data in a vector store.

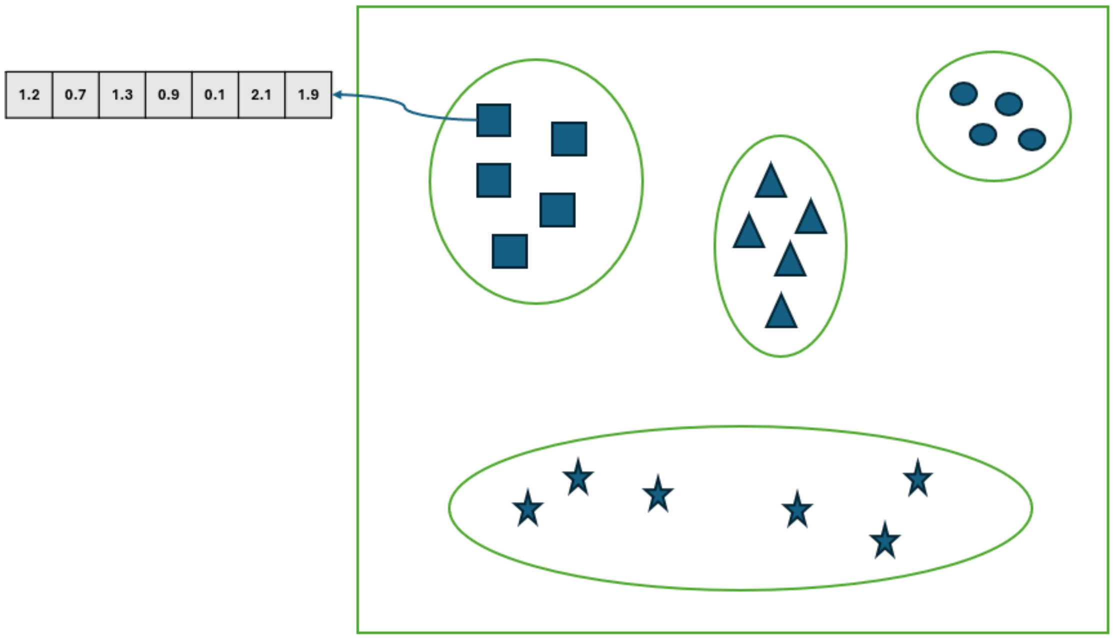
Vector store is a database that is specifically created to store vector embeddings. It uses the mathematical properties of the embeddings to store the data efficiently.

While storing the embeddings in a vector store, the indexing technique ensures that similar embeddings are organized together and dissimilar embeddings are separated using advanced algorithms like Inverted File[37] (IVF) and its variants. This helps in fast and efficient similarity search and retrieval of information in the RAG step of LegalRAG.

In LegalRAG, an in-memory vector store[38] that uses Inverted File as its indexing technique provided by LlamaIndex is used. Inverted File (IVF) is essentially using K-means to form clusters of embeddings that are similar.

Here is a sample image of how the vector store index would look like where similar embeddings are organized together and dissimilar ones are separated. Each of the geometry shown below (square, triangle, star and circle) are numerical vector embeddings. The oval shape shows the subspaces that are similar in the vector store.

Figure 3: Sub spaces of embeddings stored in vector store



Benefits of storing data in vector store:

1. Vector stores are most suitable for storing numerical vectors
2. Makes it easy to search the vector store through indexing
    a. Saves the data in a tree-like structure where each branch is a subspace
    b. When a user query searches through this tree-like structure it's easier to pass through the structure to reach the subspace which has the highest similarity to the user query.

Chapter 5 - Intent Classification

Intent classification is the integral part of the LegalRAG and this is where the novelty of the project lies. The task of intent classification is to take a user query as input and give out one of the 5 intents as the output. The five intents are: Divorce, Consumer protection, Copyright, Inheritance and Other. Therefore intent classification is a multi-class (five-class) classification problem.

DistilBERT was chosen as the model for intent classification and was fine-tuned on a dataset with 250 records or user queries. Each of the records or queries had a label or intent associated with it. 50 records were from each of the 5 intents. The dataset was divided into a ratio of 4:1 to form both training and testing dataset.

More information on the dataset can be found in the appendix of this chapter.

Pre-trained version of distilBERT is used: distilbert-base-uncased
Parameter size: 44M parameters (40% in size of BERT base)

Reason for the choice of DistilBERT:

1. The model is open-source
2. It is smaller in size when compared to other BERT models
3. Even though there are other BERT models which are bigger in the parameter size and therefore may perform better when tuned well, distilBERT is still chosen as its smaller in size and the performance it showed on the test set was extremely good as can be seen in this chapter and the appendix that it was decided that bigger models are not needed for a marginal improvement in performance. Marginal improvement brought by BERT base can also be seen in the appendix.
4. It is a transformer based model and therefore superior to traditional intent classifiers based on results from multiple research papers.
5. distilBERT is trained on multiple tasks and already has an understanding of the English language, so just fine-tuning is required. Therefore a dataset with a smaller size is enough for fine-tuning.

DistilBERT:

Distil BERT (distilbert-base-uncased) is distilled from BERT (bert-base-uncased) using a process called knowledge distillation[39] or teacher-student teaching. Knowledge distillation is a process where a teacher (larger model, in this case BERT base)’s knowledge

and generalization capabilities are transferred or compressed into a student (smaller model, in this case distilBERT).
DistilBERT has the same architecture as BERT base except that it is smaller in size, therefore has half the number of layers as BERT base, so 6 layers. DistilBERT also doesn't have few layers like token type embeddings and pooler layers. It is also trained on the same text data as BERT.

Both BERT and DistilBERT have the transformer architecture. Transformer architecture have two main components:

1. Encoder – used for natural language understanding
2. Decoder – used for natural language generation

The decoder part of the transformer architecture is clearly explained in the Large Language Models chapter.

However, BERT and DistilBERT are both Encoder-only architectures.

Encoder architecture:

Encoder architecture contains multiple parts:

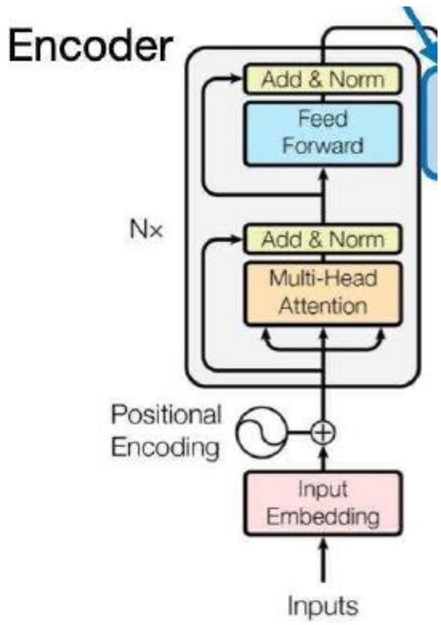
- Input embedding
- Positional Encoding
- Multi-head self-attention
- Layer Normalization and Residual connections
- Feedforward Neural network
- Followed by Layer Normalization and Residual Connection
- Softmax layer

All these layers are explained in detail in the chapter on Large Language Models. The main difference between encoder and decoer happens in the Multi-head self-attention layer.

In case of decoder, masked multi-head attention is used, where tokens can only attend to itself and the tokens that came before it, so there is a causal chain. In other words, this masked multi-head attention tries to understand the importance of all the previous tokens (including the current token) with respect to the current token.

Whereas in decoder, multi-head attention is used, where tokens can attend to itself and all the other tokens, whether they came before the current token or after it. There is no causal chain, it's completely parallel process. Unlike in masked self-attention, the tokens that come after the current token will not be masked during the self-attention calculations. The term Birectional in BERT and DistilBERT is due to this, where a token is able to get context from before and after it.

Figure 4: Encoder architecture

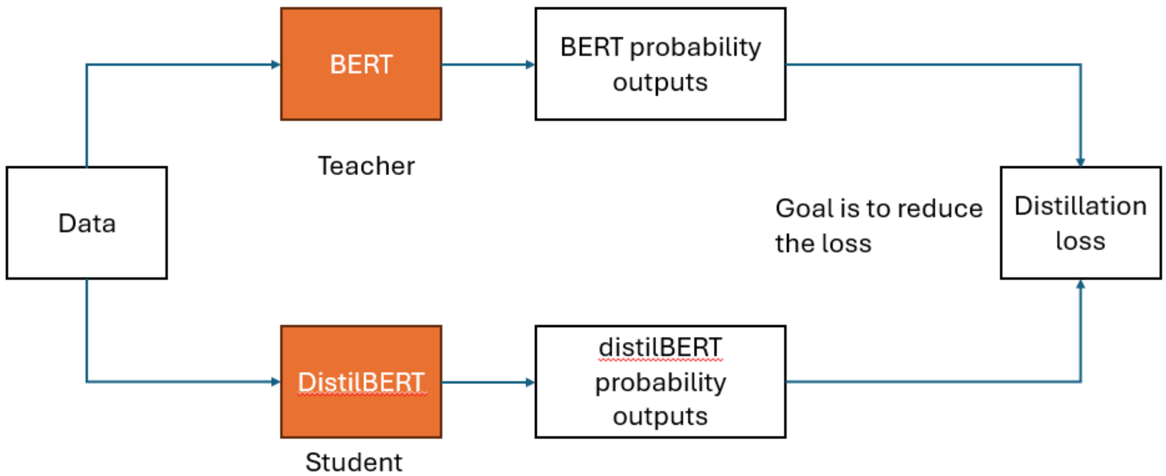


The last layer softmax is not shown in the above architecture. In case of BERT and DistilBERT, after the last Layer Normalization and residual connection (shown as Add& Norm) there will be a softmax layer.

DistilBERT training:

During the distillation process in training, the cross entropy loss was used to ensure that distilBERT's output distribution will be similar to that of the output distribution of BERT base so it will have similar generalization capabilities. In other words, instead of training distilBERT on hard labels for queries, it was trained on the BERT's probability outputs (soft labels). In a way, distilBERT is trying to mimic BERT and learn the representations of BERT.

Figure 5: Knowledge Distillation



This paper will have more information on the results of the distillation process[24].

During training, masked language modelling was considered for BERT base's knowledge to be compressed into distilBERT.

distilBERT was specifically built for its smaller size and comparable generalization capability in natural language understanding so that it can be used in applications that can be downloaded into a smartphone or IoT devices. This makes it suitable for LegalRAG which is ideal to be downloaded as an application in mobile phones..

Novelty brought by the intent classification model on the LegalRAG:

If a new legal act, for example, legal act related to corporate laws in India has to be added to LegalRAG, then it would add another data preprocessing step and a new vector store. However, due to the intent classification model, there is no update required to be made to the retrieval techniques or to the LLM. Fine-tuning an LLM requires a lot of computational resources and memory which is not always suitable.

With the intent classification model, only the distilBERT needs to be fine-tuned to do 6-class classification, with the new intent related to corporate law. Fine-tuning distilBERT which is smaller in size doesn't require much computational resources or memory.

This makes adding more legal acts in the future much easier with only few updates to the architecture.

Intent classification also makes transferring this architecture to similar problems much easier. For example, if there was a need to build a science based system that answers user queries related to classical physics, astronomy, chemistry, and biology. Same architecture as LegalRAG can be used. Instead of legal data, the vector stores will contain scientific data. The chunking and retrieval techniques may change based on the size of the scientific data. However, no update to the LLM is required, no fine-tuning required. Only the intent classification model, distilBERT will have to be fine-tuned to route the scientific queries to the relevant vector store. There is the similar benefit of using lesser resources and memory for fine-tuning a smaller intent classification model than a larger LLM.

Chapter 6 - Large Language Model

A Large Language Model or LLM is used to generate the responses for the user query in LegalRAG. Zephyr Alpha[40], an LLM with 7 billion parameter size is used in LegalRAG.

Reason for the choice of Zephyr Alpha:

1. It is open-source
2. The model size is 7 billion parameters, which is less compared to other LLMs that are much bigger in size. Therefore this model is less likely to cause memory issues and requires slightly less compute power.

Some of the evaluation metrics like Context Relevance Evaluator and Faithfulness Evaluator which will be described in the next chapter also require an LLM. Zephyr Alpha cannot be used there because Zephyr Alpha would always inform that the context and the response are relevant and faithful as it is generated by the same LLM. Instead a different LLM is used for the evaluation. Llama3-8b-instruct[41] is the evaluation LLM. The reason for choosing this LLM is the same as the ones present for Zephyr Alpha.

This chapter provides the workings of Large Language Models.

LLMs are made up of transformer architecture which was introduced in this paper. The paper introduces the encoder-decoder architecture of transformers, encoder is for natural language understanding and decoder is for natural language generation. Most of the LLMs only contain the decoder part of the transformers because that's the part needed for generating text.

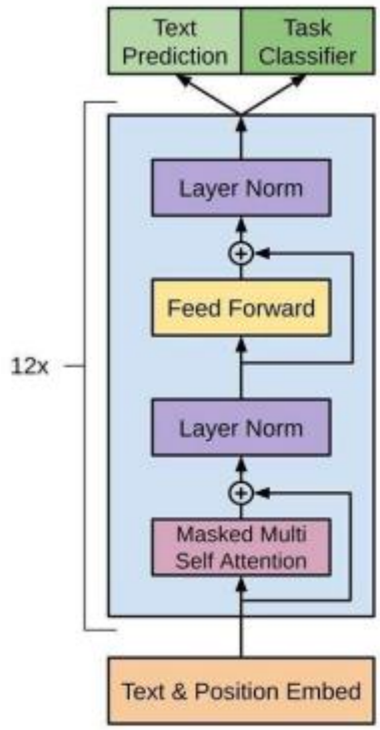
In brief, the output of the decoder architecture is to predict the next word/token given the previous words/tokens.

Each LLM's decoder architecture will have multiple stacked transformer layers leading up to 7 billion parameters for Zephyr Alpha and 8 billion parameters for Llama3-8b-instruct. For larger LLMs like GPT 3.5, there are about 175 billion parameters.

Decoder architecture

The following architecture taken from Attention is All you Need[42] paper shows the decoder architecture of the LLM. In the paper 12 transformer layers were included in the decoder, but in LLMs there will be many more layers:

Figure 6: Decoder architecture



These are the components of the decoder architecture:

1. Text embedding
2. Position embedding
3. Masked multi-head attention
4. Layer normalization
5. Feed forward neural network
6. Output

Text embedding:

LLMs generally use byte-pair encoding[43] to embed the words or token into numerical vectors. Byte pair encoding starts with individual characters. It checks which pair of characters appear together very frequently in the training corpus. Pairs those two characters that appear frequently into a single token. This new token gets an ID. It repeats the process until all tokens get assigned to an ID.

Example: Let's take the word 'Hello'. Initially each character in the word 'hello' would be given an ID

‘H’ -> 1
‘E’ -> 2
‘L’ -> 3
‘L’ -> 3
‘O’ -> 4

Let’s say in the training corpus, pairs of characters appear in the following frequency

‘He’ appears 10,000 times

‘Ll’ appears 1000 times

Since the pair ‘he’ appears frequently together, it becomes a new token and gets assigned a new ID. For example:

‘He’ -> 5

‘Ll’ -> 6

This process continues where the new tokens ‘he’ and ‘ll’ are combined and seen how many times ‘hell’ appears together and given a different token. This process repeats until every pair of characters and words are given unique IDs.

Advantage of using byte-pair encoding:

1. Use both sub-words and words tokens balancing both character level and word level tokens
2. Even unknown words gets broken down into familiar sub-words and get assigned IDs

Position encoding:

Word embeddings like byte-pair encoding or other methods like word2vec don’t consider word order. Positional encoding[42] is important because transformers process words parallelly, so there needs to be something to include the word order.

The following two sentences would be treated the same even though the meanings are different without word order:

1. I run every morning
2. Every morning run I

Positional encoding will provide a numerical vector which has the same size as the text embedding. Each element in this positional encoding vector is generated by a sinusoidal function (sine or cosine). This sinusoidal function takes in as its inputs: the position of the word in the sentence (pos), dimension of the text embedding (d) and the index in the embedding dimension (i).

The paper 'Attention is all you need' used the following sinusoidal functions to get the positional vector:

Figure 7: Sinusoidal functions that make up Positional Encoding

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

The word embedding is obtained by adding the text embedding and the positional encoding

Embedding = text embedding + positional encoding

The above addition is an element wise addition of two vectors of the same dimensions.

Masked multi-head attention:

Masking:

Masking is done to ensure that in the calculations of multi-head attention for a given token, only the tokens that come before it will be used. No information from the tokens that come after it will be used as that would be a data leakage.

The tokens that come after the current token are masked so that in the calculations only the current token and its previous tokens are used.

Attention:

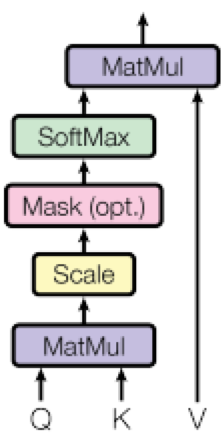
Before getting to multi-head attention, let's understand self-attention. Generally when the context of a word is considered, all the words in the context window are given equal weightage or importance. However, that's not always true in reality, some words are more important than others. Attention is a mechanism that tries to quantify the importance of the other words with respect to the given word.

In the transformers architecture, there are three numerical vectors that are generated from the word embeddings that are sent as input to the transformers. These three vectors are Query, key and value. These three vectors are generated by multiplying the input embedding with 3 matrices. The weights of these matrices Wq, Wk and Wv are obtained during training.

1. Query vector (Q) - this is the vector that queries the surrounding words
2. Key vector (K) - This is the vector that acts as the reference for the surrounding words
3. Value vector (V) - This is the actual information needed from the surrounding words to calculate the attention scores

The following architecture shows the self-attention calculation:

Figure 8: Self-attention architecture



The dot product of Query and key gives the attention weights. These weights tell how each token or word attends to the other tokens in the sentence. Higher the weight implies higher the importance of the word on the query token.

Note that this attention also includes identifying the importance of the given token itself, hence the name self-attention.

Mathematically it will look like this:
Attention score = softmax(dot product of Q and K) * V

After the dot product is taken usually, the dot product is divided or scaled by the square root of the dimension of the key vector. This is to ensure that the dot product doesn't have a high magnitude and go out of bounds due to the high dimensionality of the key vector.

In summary, the goal of self attention is to get the importance of the surrounding words on the given word.

Multi-head attention:

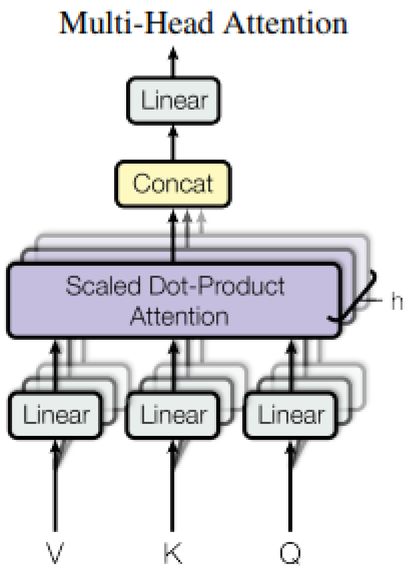
Self-attention captures one type of relationship between words. There could be many relationships or different aspects of meanings between words. Multi-head means multiple sets of self-attention mechanisms that happen parallelly. Each of the head can focus on different aspect of the meaning. For example for a 3-head multi-head attention:

1. Head 1 focus on the action in the sentences
2. Head 2 focuses on the grammatical relationships
3. Head 3 focuses on the structure of the sentence

Multi-head attention formulas remain the same as the single head attention

Following architecture shows the multi head attention architecture with h heads:

Figure 9: Multi-head attention architecture



The Linear layers at the bottom are the 3 matrices Wq, Wk and Wv. Each head will have a different set of Wq, Wk, and Wv. There are h number of Scaled dot-product attention mechanisms (shown in the previous figure). The results of all h heads are concatenated in the end and the final linear layer is another matrix multiplication that reduces the dimension of the output of the multi-head attention layer. This ensures that output dimension of all multi-head attention layers are the same and doesn't lead to the dimension to keep increasing as we move from one transformer layer to the next.

Layer normalization:

Layer normalization is required to speed up the training process and to ensure the activations (output of multi-head attention) is normalized. This leads to stability and prevents the activations from blowing up.

The following shows what happens in the layer normalization layer:

Output activation = (input activation - mean)/standard deviation * gamma + beta

The first step is subtracting the activations by its mean and dividing by the standard deviation to make the mean go to 0 and standard deviation to 1. Then scaling the standard deviation to gamma and mean to beta. Gamma and beta are learnable parameters.

Feed forward layer with residual connection:

Output of feed forward layer = feed forward layer(normalized activations from layerNorm) + normalized activations from layerNorm

Feedforward layer brings the non-linearity into the picture. Normalized activations from the previous layers are sent as input to the feedforward layer. Each token is independently processed by the feedforward layer. There is no interaction between the tokens as seen in multi-head attention layers.

Residual connections[44] ensure that normalized activations flow directly to the output of the feedforward layer and prevent any vanishing gradients issue that is common when a large number of transformer layers are stacked one after the other.

Feedforward layer is followed by another layer normalization layer to normalize the activations from the feedforward layer and to stabilize the training.

The output of the feedforward and layer normalization will be the hidden representation of each token.

Softmax activation:

The final layer in a transformer decoder architecture is a softmax function that takes the output from the feedforward layer and produces the probability of the next word or token given the previous words.

Following shows the formula for softmax function:

Figure 10: Softmax function

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Here z_i is the hidden representation of each token.

Training

Given the transformer decoder structure and all its components, let's focus on the training process of LLM. The training process and datasets used differ from LLM to LLM. Following steps show what is generally followed in some of the open source LLMs[45]:

1. Train a supervised policy
2. Train a reward model
3. Optimize the policy against the reward model using reinforcement learning

Training a supervised policy:

In this step, samples are taken from a large dataset of prompts. A human demonstrates the correct answer to that prompt or question. The question and the human response is used to fine-tune the decoder transformer of the LLM.

This method of demonstrating the correct responses to different questions is also referred to as Instruction tuning[46].

This is supervised learning as there is a human being giving the correct responses to fine-tune the LLM.

Train a reward model:

Here prompts are sent to the model and the LLM outputs several responses for the same prompt or question. All the responses are given to a human who then ranks all the responses from best to worst. The human ranking for the answers for all the prompts sent to the model is used to train a reward model.

For example, for a prompt, if the model gave 3 responses: response A, response B, response C, then the human ranking may look like this:

Response B > response A > response C
The ranking indicates that response B is better than response A which in turn is better than response C.

This is commonly referred to as Reinforcement Learning from Human Feedback[47] (RLHF)

Optimize the policy against the reward model using reinforcement learning:

Prompts or questions are sampled from the dataset and sent to the LLM. LLM generates a response using its supervised policy. The response from LLM for the given prompt is given a reward by the reward model. The reward is used to optimize the supervised policy.

This technique is commonly referred to as Proximal Policy Optimization[48] (PPO).

The issue with PPO is that it requires training the policy and the reward model. There is a need for human evaluation in both steps which is a costly process.

Direct Preference Optimization:

To avoid training two different models, a new method called Direct Preference Optimization[49] (DPO) is used.

In this method, the first step is the same as before, i.e., creating a supervised policy. Then sample questions or prompts from the dataset and the model will produce several responses for each of the prompt. A human will create a preference pairs. For example: If model produced response A, response B and response C for a prompt, then a human may give the following preference pairs:

Response A > response C
Response B > response A
Response B > response C

Here the preference pairs indicate that response A is a better response than response C, and response B is a better response than response A.

These human preferences along with the prompts are used to optimize the policy that intrinsically builds a reward function, so there is no need for an additional reward model.

So, in the case of DPO there is only one model to train and optimize.

The above steps describe the general steps involved in training an LLM that could be used to provide natural language response to textual query.

Hallucination:

Hallucination[50] is a term that refers to the cases when an LLM provides a response to the query that is incorrect. It is a case of LLM making things up instead of stating that it doesn't have the information to answer that particular question.

Hallucination is an important problem to address in LegalRAG because it is important that users get correct answers to their legal queries and the consequences of 'made-up' hallucinated responses may be grave.

Hallucinations can happen for many reasons:

1. Query requires knowledge that was not learned in LLM's training
2. Model generalizes its training knowledge to domain specific knowledge incorrectly
    a. For example: LLM may use information from U.S legal system to India
3. Retrieved context doesn't contain the information to answer the question.

To check for hallucination, there is a faithfulness evaluator metric used to ensure LegalRAG's responses are accurate and not made-up.

Chapter 7 - Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) is one of the best ways to combine the informal retrieval capabilities with the generation capabilities of the LLM. It is a way of providing domain knowledge to the LLM. The domain knowledge may or may not have been part of the training set of the LLM. This is especially useful when the domain knowledge is not publicly available information and specific to an organization. In such cases, the LLM would definitely not have access to such non-public data during its training and would not perform well when queries are asked related to that. This would commonly lead to hallucination.

The next chapter describes LLM and hallucination in detail.

One way to resolve this issue is to fine-tune the LLM with this non-public domain knowledge which would improve LLM's performance when asked questions related to this. However, fine-tuning LLM is a costly process as it requires adding more memory and requiring more computational resources. If done incorrectly, the LLM may lose the information it had learned during its training. Even with fine-tuning, there is no guarantee that the LLM will perform better on this non-public information.

Another flaw of fine-tuning LLMs is observed when the non-public information gets updated quickly. This would require fine-tuning the LLMs repeatedly so that the response provided by the LLM is up-to-date. This is a very costly process.

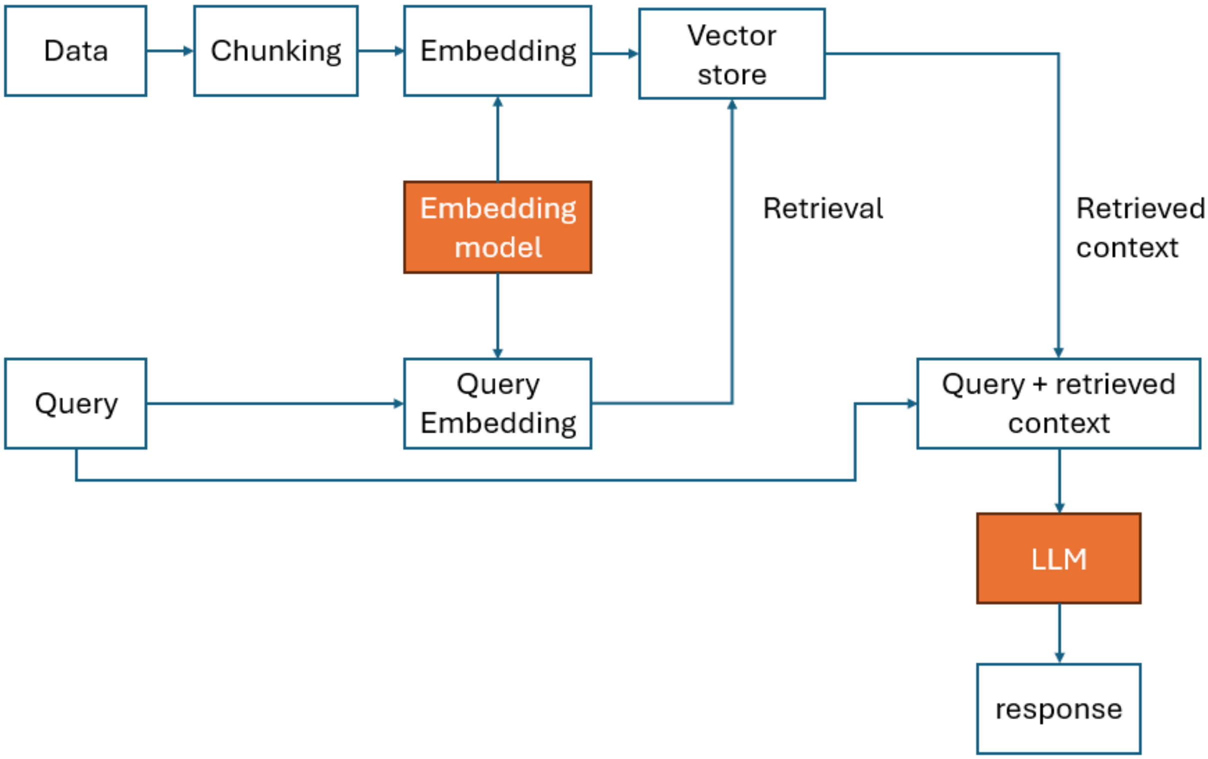
RAG avoids fine-tuning the LLM and by combining the retrieval technique which retrieves relevant context from this non-public information using the query and shares the context to the LLM which can use both the query and the retrieved context to provide responses that are more accurate. RAG also provides source verification support on the responses it provides, as it provides the document name, page number and other metadata that would help anyone verify the response provided by the LLM.

RAG has 3 components:

1. Ingestion
2. Retrieval
3. Generation

Below is a simplified architecture of the RAG:

Figure 11: RAG architecture



As mentioned earlier, Chunking, embedding and storing the data in vector store are part of RAG but explained in this document as a part of data preprocessing

Ingestion component of RAG involves taking in the data. Chunking the data into smaller parts or sentences. Chunked data is embedded into a numerical vector using an embedding model. In this case the embedding model used is bge-small-en-v1.5 (as explained in Data preprocessing step). After the numerical vector embeddings are created, data gets stored in the vector store.

Chunking, embedding and vector stores are explained in detail in the Data preprocessing chapter.

Retrieval component of RAG includes embedding the query and searching the vector store to get the relevant context.

Everytime a user asks a textual query. The query gets converted to a numerical vector embedding using the same embedding model as the one used after chunking, i.e., bge-small-en-v1.5

If the same embedding model is not used then the query embedding and the embeddings stored in the vector store may have different lengths and the values in the embeddings will have different numbers and encode different information, so searching the vector store will not be possible.

After the query is embedded it searches the vector store, and the search is made easy as the vector store would have indexed the data in such a way that similar embeddings would have similar embeddings and indexed similarly.

In a simple retrieval technique, query embedding would search through the vector store and give out the most similar context from the vector store. Most similar context is decided by the cosine similarity in general. However other similarity measures can also be used.

This chapter will include more description on some of the advanced retrieval techniques and the result comparison between the different retrieval techniques can be seen in the Results chapter and in the appendix.

The final component of RAG is Generation. The retrieved context from the retrieval process and the query is sent to the LLM as inputs. LLM generates a response that answers the query. The user can have additional verification on the sources by looking at the retrieved context and the document and page numbers from which the response was taken.

More information on the LLM is provided in the next chapter.

In LegalRAG, there are four of these RAG systems, one for each of the legal acts.

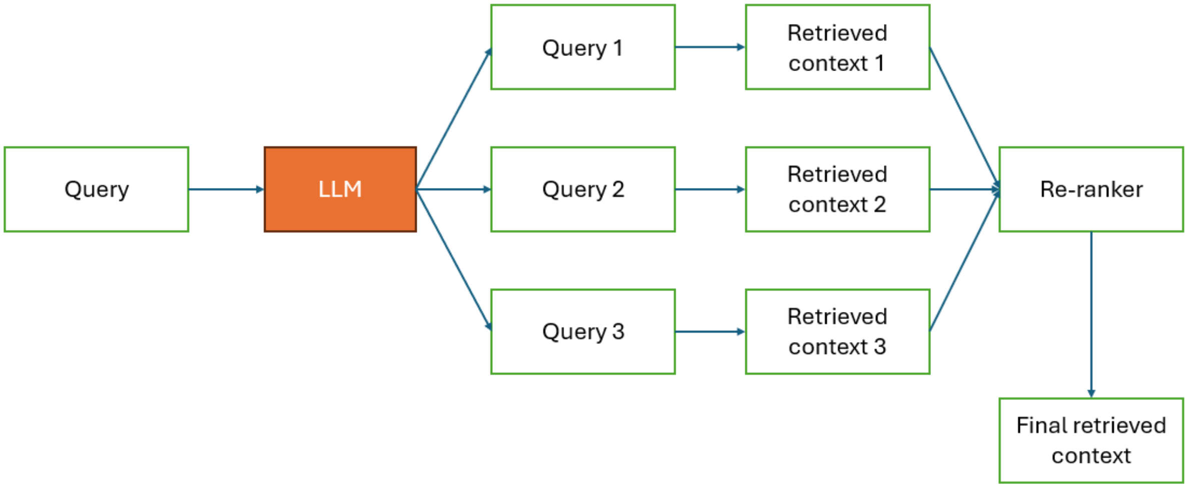
Some of the advanced retrieval techniques used in RAG are:

1. Query fusion
2. Hypothetical Document Embeddings (HyDE)

Advanced retrieval techniques help in retrieving context that may be more relevant to the query than the simple cosine similarity based retrieval technique.

Query fusion:

Figure 12: Query fusion architecture



The above architecture shows how Query fusion works. It takes a user query and passes it through an LLM. The LLM is made to generate multiple variations of the original query which is shown as Query 1, Query 2 and Query 3 in the figure. Each of these queries are embedded and searched through the vector store and relevant context is retrieved. The relevant context from all 3 queries are passed through a re-ranker whose job is to find the top relevant context from all the retrieved contexts.

Usually the re-ranker[51] is a cross-encoder[52] or BERT which takes in the query and the retrieved context and provides a similarity score based on deep semantic understanding rather than a simple cosine similarity.

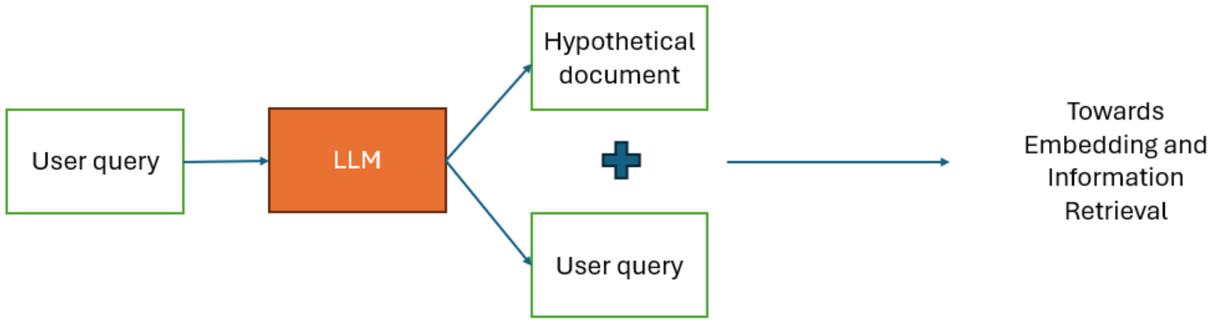
It should be noted that the embedding of each of the sub-queries is created by the same embedding model that was used for embedding the data stored in vector store, i.e., bge-small-en-v1.5

Query fusion is usually done when the question is long and has many aspects to it. With the help of query fusion different aspects of the question can come out as the sub queries and those subqueries which are more focused on certain aspects can retrieve contexts that are more relevant to those aspects. When combined together and re-ranked, the chances are high that the top context is much richer than the case where the original query was sent for retrieval.

LLM used to generate these variations of the query can be the same one or a different one than the one used in response generation. In LegalRAG, the same LLM has been used for Query fusion and final response generation.

Hypothetical Document Embeddings (HyDE):

Figure 13: HyDE architecture



HyDE is a method where the user query passes through an LLM and the LLM generates additional information about the user query. This additional information called a hypothetical document can be accurate information or hallucinated information. It is not a problem even if it is hallucinated as the purpose of HyDE is to provide additional context about the user query or a way of enriching the user query.

Some user queries may not be detailed and if just the user query is sent to search the vector store it may not get the most relevant context. So enriching it with a hypothetical document will be useful.

The hypothetical document is embedded using the same embedding model (bge-small-en_v1.5) that was used to embed the data in while storing in the vector store.

The LLM used in HyDE can be the same LLM that is used in the Generation component of LegalRAG or a different one. The same LLM has been used here.

In the chapter on Results, evaluation will be done on all 3 retrieval techniques and in the appendix the results on specific questions can also be seen for all 3 retrieval techniques.

Chapter 8 - Evaluation metrics

This chapter describes all the evaluation metrics that were considered to measure the performance of LegalRAG for various user queries that covers all 4 legal acts under consideration.

More information on the evaluation dataset can be found in the appendix of this chapter

Following are the evaluation metrics considered for evaluating Intent classification:

1. Precision
2. Recall
3. F score
4. Accuracy

Since the intent classification model is a multi-class (5-class) classification, the above metrics will be calculated for each intent and then averaged in the end to get a single value for the metrics. In other words, it will be treated as a binary classification with the selected intent as class 1 and all other intents as class 0 and the metrics will be calculated. After doing this 5 times (once per each intent), the results will be averaged.

Following are the definitions of the terms used in the evaluation metric:
True positive = this is when the model says its class 1 and it is actually class 1
True negative = this is when the model says its class 0 and it is actually class 0
False positive = this is when the model says its class 1 but it is actually class 0
False negative = this is when the model says its class 0 but its actually class 1

Precision:

Precision tells the percentage of records where the model accurately predicted class 1 out of all the records it predicted as class 1.

Precision = True Positives/(True positives + False positives)

Recall:

Recall tells the percentage of records where the model accurately predicted class 1 out of all the records which were actually class 1

Recall = True Positives/(True positives + False negatives)

F score:

F score is the harmonic mean of precision and recall. F score will only be high when both precision and recall is high. It will be low when either precision or recall is low. F score is a useful metric when measuring and reducing false positives and false negatives are equally important

F score = 2 * Precision * Recall/ (Precision + Recall)

Accuracy:

Accuracy is a measure of how many of the intents did the model get correct out of all the records in the dataset.

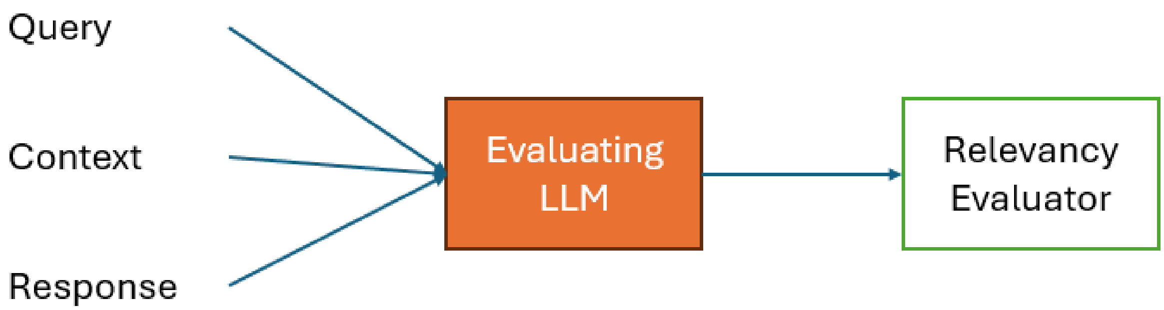
Accuracy = (True positives + True negatives)/(True positives + True negatives + false positives + false negatives)

Following are the evaluation metrics considered for the RAG section of LegalRAG:

1. Relevancy Evaluator [31]
2. Faithfulness Evaluator [32]
3. Semantic Similarity Evaluator [53]
4. Correctness Evaluator [54]

Relevancy Evaluator:

Figure 14: Relevancy Evaluator architecture



The above architecture takes user query, context and the generated response as its inputs. It uses an Evaluating LLM to check whether the context retrieved and the provided response matches the user query or not.

Relevancy Evaluator is an LLM based evaluator that tells us how good the retrieval and generation mechanism is. If the mechanisms are not good enough that means the retrieved context and the generated response is not relevant to the user query and this is shown by the relevancy evaluator.

Evaluating LLM has to be a different one from the one used to generate responses in the RAG systems. It cannot be the same one as it may always indicate that the queries, contexts, and responses are associated and the user query was answered by the response generated.

The Evaluating LLM used for LegalRAG is an 8 billion parameter model, open source model available in HuggingFace: Llama3-8b-instruct

The task of Evaluating LLM is to evaluate whether the user query matches or is answered by the generated response and the retrieved context. Even similarity measures or keyword based matching can be used for this evaluation, but an LLM is used to make the evaluation. LLM evaluation is generally more accurate than similarity measures and keyword based matching, but may take longer to produce a result.

LLM evaluation happens by giving a systemic prompt to the LLM instructing it to compare the query to the context and response to see if they match the user query or not.

The output of Relevancy Evaluator will be True if the response and context matches the user query. It will be False if that is not the case. It also provides textual feedback

Faithfulness Evaluator:

Figure 15: Faithfulness Evaluator architecture

Faithfulness Evaluator is a hallucination check, where the evaluating LLM will take the retrieved context and the generated response and confirm if the response came from the context or not. Faithfulness Evaluator is True if there is no hallucination, False if there is hallucination. It also provides textual feedback.

Unlike the relevancy evaluator which also tests the retrieval part, faithfulness evaluator checks the LLM's generation capability to provide a relevant response. If the LLM is not responding based on the retrieved context then it means the LLM is hallucinating and is providing a response that is not true.

In order to do the evaluation of whether the response matches with the retrieved context or not, the faithfulness evaluator uses an evaluating LLM.

Llama3-8b-instruct is used as the Evaluating LLM for Faithfulness Evaluator as well.

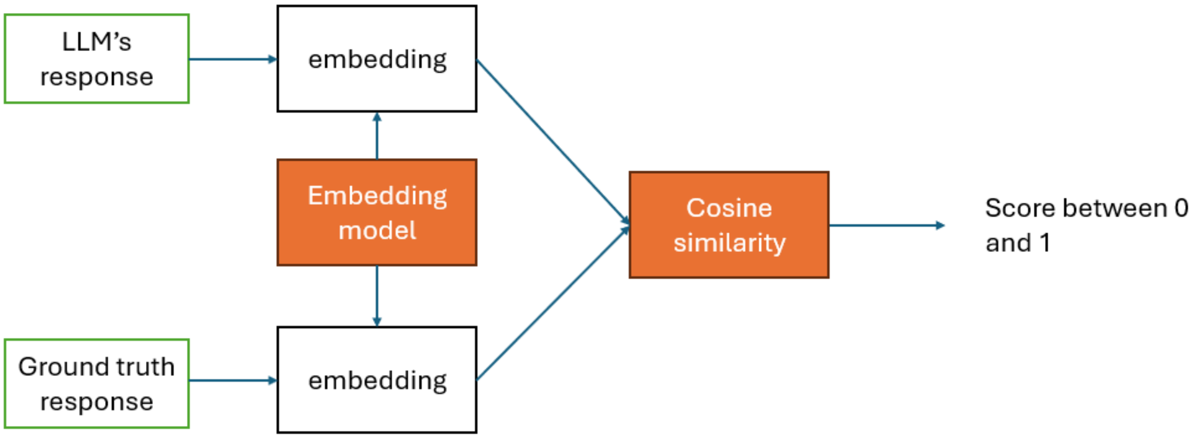
LLM evaluation is generally more accurate than other methods like similarity search and keyword based search but it is also computationally more intensive and may take longer to provide the result.

LLM evaluation happens by giving a systemic prompt to the LLM instructing it to compare the context and the response to see if the response seems to be taken from what's in the context or if there are any new facts that are not in the context.

Semantic Similarity Evaluator:

Semantic Similarity Evaluator provides a score between 0 to 1.

Figure 16: Semantic Similarity Score architecture



The task of semantic similarity evaluator is to compare the legalRAG's LLM's generated response for a user query with a reference response manually provided by a rational human being.

An evaluation dataset has been created with user queries and reference responses (or ground truth responses) so legalRAG can respond to the user queries in the evaluation dataset and the LLM's responses can be compared with the ground truth responses.

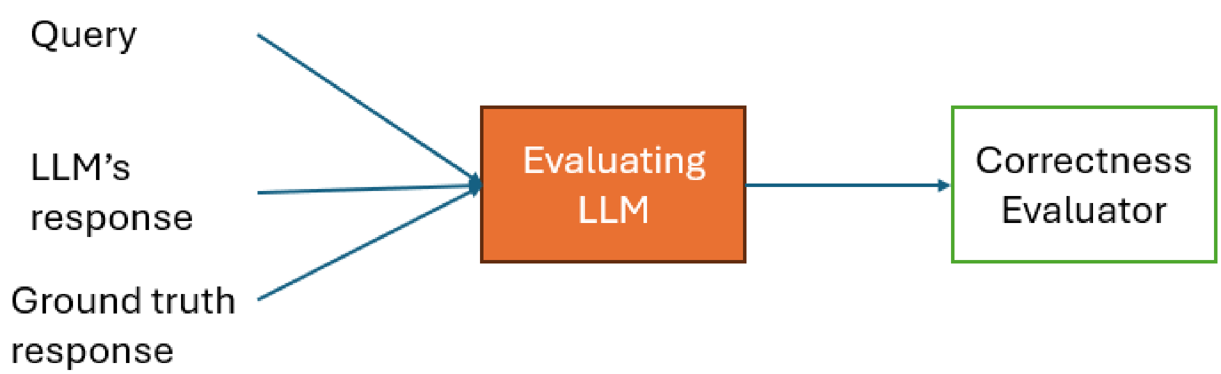
To compare the LLM's response with the reference response, Semantic Similarity Evaluator uses an embedding model to convert both LLM's response and the reference response into a numerical vector embedding. Cosine similarity is used to get how similar the two responses are. That's why this evaluator is between 0 to 1. Generally, if cosine similarity is greater than 0.8 that's a pass (LLM response and reference response are very similar else it's considered as fail (LLM response and reference response are dissimilar).
The embedding model used here is same as the one used to convert the legal data into embeddings before being stored in the vector store: bge-small-en-v1.5

A different embedding model could also be used, but bge-small-en-v.15 was chosen for its performance.

Correctness Score:

Correctness Evaluator provides a score between 1 to 5. Generally, the threshold is 4 i.e., if the score is greater than 4 it's a pass, else it's a fail.

Figure 17: Correctness Evaluator architecture



Correctness evaluator checks the relevance and correctness of the legalRAG's LLM response with a reference response (or ground truth response) and the user query.

An evaluation dataset has been created with user queries and reference responses (or ground truth responses) so legalRAG can respond to the user queries in the evaluation dataset and the LLM's responses can be checked for relevance and correctness with the ground truth responses and the user query.

Unlike Semantic Similarity score, this doesn't look at embeddings and cosine similarity. Correctness Evaluator uses an LLM based evaluation to check whether the LLM response is similar to the reference response or not.

LLM evaluation happens by giving a systemic prompt to the LLM instructing it to compare the LLM's response with the ground truth response and the user query and verify the relevance and correctness of the LLM's response.

Llama3-8b-instruct is the LLM used for this evaluation.

The above 4 evaluation metrics as it measures all aspects of the RAG systems in LegalRAG:

1. Relevancy evaluator tests the retrieval and generation part of LegalRAG

2. Faithfulness evaluator tests the LLM’s response generation capability as well as ensures that LLM is not hallucinating and generating wrong response
3. Semantic Similarity score evaluator and correctness score tests how well the overall response of the LLM is when compared to the ground truth response that’s manually picked by a human expert.

Chapter 9 - Results

This chapter describes the results obtained on intent classification and RAG systems and the overall LegalRAG system.

Results for intent classification:

distilBERT was the model chosen for intent classification. A dataset was created that contains 250 user queries along with the intent label. There were 50 queries per intent. The dataset was divided into 4:1 for training and testing.

The model was fine-tuned for 5 epochs, below are the results obtained on training and testing for distilBERT 5-class classification intent recognition:

Table 1: DistilBERT performance

Metrics	On training dataset	On testing dataset
Recall	100%	96%
Precision	100%	96.36%
F score	100%	95.98%
Accuracy	100%	96%
Loss	0.02	0.07

It can be seen from the above results that on the training dataset, distilBERT is getting all the intents correctly. From testing set results it can be seen that distilBERT's class classification model seems to perform very well on unseen data. There is no overfitting as both the training and testing metrics and loss are very close to each other.

Results on BERT base are shown below, to justify the reason for choosing distilBERT, a smaller model when compared to BERT base, a relatively larger model with more generalization capability.

Table 2: BERT performance

Metrics	On training dataset	On testing dataset

Recall	100%	100%
Precision	100%	100%
F score	100%	100%
Accuracy	100%	100%
Loss	0.014	0.03

BERT base has 100% on both training and testing set, showing that it performs better than distilBERT. However, it should be noted that this is just a marginal 4% improvement in performance.

DistilBERT was chosen because its smaller in size, marginally worse than BERT base, but has very fast inference, unlike BERT base which has better performance but takes longer for making inferences. Fast inferences are better in LegalRAG because intent recognition is only a small part of the project and there are further steps that need to be accomplished to get the user response. Slow inference with BERT base can affect the overall speed to get the user response, also BERT would be a burden on the memory. All these make the choice of distilBERT to be optimal.

Results for the RAG system and the overall LegalRAG system:

An evaluation dataset was created with 6 user queries per intent. For each query reference context and reference response was provided. These reference contexts and reference responses will be treated like the ground truth. Overall results of the LegalRAG and the RAG system for the metrics described in the previous chapter is shown below.

Results are separated based on the retrieval mechanism. The retrieval mechanism that led to the overall best results will be chosen.

Below tables show the results of the Relevancy Evaluator and Faithfulness Evaluator for each of the four legal acts. As explained in the previous chapter, these evaluators provide a True/False response. The table below shows the percentage of records where the results were True

Table 3: Relevancy and Faithfulness evaluator results on simple retrieval of copyright act data

Metrics (for Copyright)	Simple Retrieval	HyDE	Query fusion
Relevancy Evaluator Score	100%	100%	100%
Faithfulness Evaluator Score	66.66%	66.66%	66.66%

Table 4: Relevancy and Faithfulness evaluator results on simple retrieval of divorce act data

Metrics (for divorce)	Simple Retrieval	HyDE	Query fusion
Relevancy Evaluator Score	60%	80%	100%
Faithfulness Evaluator Score	40%	40%	20%

Table 5: Relevancy and Faithfulness evaluator results on simple retrieval of consumer protection act data

Metrics (for Consumer protection)	Simple Retrieval	HyDE	Query fusion
Relevancy Evaluator Score	100%	100%	100%
Faithfulness Evaluator Score	60%	60%	80%

Table 6: Relevancy and Faithfulness evaluator results on simple retrieval of inheritance act data

Metrics (for Inheritance)	Simple Retrieval	HyDE	Query fusion
Relevancy Evaluator Score	100%	100%	100%
Faithfulness Evaluator Score	80%	80%	100%

Query fusion has 100% relevancy for all 4 legal acts. Although it has lower faithfulness for divorce act, it has equivalent faithfulness in the other 3 acts. Since, Query fusion performs the best in all the acts except divorce. Query fusion can be considered the best retrieval mechanism for LegalRAG. But, it should be noted that this was just a marginal improvement brought by Query fusion

The results for Semantic Similarity Evaluator and Correctness Evaluator are shown for each legal act in the below tables. Since these two evaluators are numerical scores, their descriptive stats are provided.

A split of the 3 different retrieval mechanisms is also shown to facilitate the choice of best retrieval mechanism for LegalRAG

Table 7: Semantic similarity and Correctness evaluator results on simple retrieval of copyright act data

Metrics (for Copyright)		Simple Retrieval	HyDE	Query fusion
Semantic Similarity Evaluator (score 0 to 1)	Mean	0.83	0.83	0.82
	Standard deviation	0.11	0.1	0.09
	Minimum	0.7	0.7	0.7
	Maximum	0.98	0.98	0.96
Correctness Evaluator (score 1 to 5)	Mean	3.83	3.83	3.83
	Standard deviation	0.4	0.4	0.4
	Minimum	3	3	3
	Maximum	4	4	4

Table 8: Semantic similarity and Correctness evaluator results on simple retrieval of divorce act data

Metrics (for Divorce)		Simple Retrieval	HyDE	Query fusion

Semantic Similarity Evaluator (score 0 to 1)	Mean	0.86	0.85	0.85
	Standard deviation	0.1	0.09	0.1
	Minimum	0.72	0.72	.72
	Maximum	0.98	0.96	0.98
Correctness Evaluator (score 1 to 5)	Mean	3.6	3.6	3.6
	Standard deviation	0.54	0.54	0.54
	Minimum	3	3	3
	Maximum	4	4	4

Table 9: Semantic similarity and Correctness evaluator results on simple retrieval of consumer protection act data

Metrics (for Consumer protection)		Simple Retrieval	HyDE	Query fusion
Semantic Similarity Evaluator (score 0 to 1)	Mean	0.87	0.89	0.87
	Standard deviation	0.04	0.04	0.04
	Minimum	0.82	0.82	0.82
	Maximum	0.92	0.93	0.92
Correctness Evaluator (score 1 to 5)	Mean	4	4	4
	Standard deviation	0	0	0
	Minimum	4	4	4

	Maximum	4	4	4

Table 10: Semantic similarity and Correctness evaluator results on simple retrieval of inheritance act data

Metrics (for Inheritance)		Simple Retrieval	HyDE	Query fusion
Semantic Similarity Evaluator (score 0 to 1)	Mean	0.82	0.82	0.79
	Standard deviation	0.13	0.13	0.08
	Minimum	0.62	0.62	0.69
	Maximum	0.98	0.98	0.91
Correctness Evaluator (score 1 to 5)	Mean	3.8	3.8	3.6
	Standard deviation	0.44	0.44	0.89
	Minimum	3	3	2
	Maximum	4	4	4

In terms of semantic similarity score and correctness evaluator, the three retrieval mechanisms show similar performance with minor differences.

So overall, Query fusion can be chosen as the retrieval mechanism as it seems to marginally outperform the other two methods based on the 4 metrics considered. However, it should be noted that all 3 retrieval mechanisms are performing very well in terms of the 4 metrics considered.

Chapter 10 - Conclusion

Project LegalRAG has met all the objectives listed in the Introduction. The main components of legalRAG like intent recognition and RAG systems are performing well as seen in the previous section. Previous section also justifies the choice of distilBERT for intent classification and query fusion for the RAG systems through performance.

It is evident from the results that overall legalRAG is providing relevant and accurate responses to user queries related to the four Indian legal acts in consideration.

Recommendations:

Even though the performance of legalRAG is good, there are a few things that would be beneficial to include to make it beneficial tool for the general public of India:

1. Build a diverse set of large evaluation dataset collected through actual public questions to get a better understanding of the sub areas within each legal act where legalRAG is not performing well so relevant information regarding those sub-areas could be included in the vector stores
2. Add more legal acts other than the 4 used for the purpose of this project for legalRAG to be truly beneficial to the users.
3. Build an app for legalRAG that would make it easy for users to access this system. This would also help in building the diverse evaluation set mentioned in the previous point. LegalRAG can also be improved through user feedback.
4. Ensure the data sources are directly connected to the Department of Justice and other judiciary websites of India so that latest updates to the law and information regarding latest legal cases can be uploaded and stored in the vector stores so that the users always get the up-to-date response regarding the legal acts.

Bibliography/References

[1] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan et al., "Retrieval-Augmented Generation for Large Language Models: A Survey", arXiv:2312.10997, URL https://arxiv.org/abs/2312.10997

[2] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, "LoRA: Low-Rank Adaptation of Large Language Models", arXiv:2106.09685, URL https://arxiv.org/abs/2106.09685

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks", arXiv:2005.11401, URL https://arxiv.org/abs/2005.11401

[4] URL https://spacy.io/api/entityrecognizer

[5] URL https://www.llama.com/

[6] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani et al., “Zephyr: Direct Distillation of LM Alignment”, arXiv:2310.16944, URL https://arxiv.org/abs/2310.16944

[7] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song et al., “DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning”, arXiv:2501.12948 , URL https://arxiv.org/abs/2501.12948

[8] URL https://openai.com/

[9] URL https://claude.ai/

[10] URL https://mistral.ai/

[11] URL https://openai.com/index/gpt-4/

[12] URL https://openai.com/o1/

[13] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan et al., “Large Language Models are Few-Shot Learners”, arXiv:2005.14165 , URL https://arxiv.org/abs/2005.14165

[14] Brian Lester, Rami Al-Rfou, Noah Constant, “The Power of Scale for Parameter-Efficient Prompt Tuning”, arXiv:2104.08691 , URL https://arxiv.org/abs/2104.08691

[15] Xiang Lisa Li, Percy Liang, "Prefix Tuning: Optimizing Continuous Prompts for Generation", arXiv: 2101.00190, URL https://arxiv.org/abs/2101.00190

[16] Clifton Poth, Hannah Sterz, Indraneil Paul, Sukannya Purkayastha, "Adapters: A Unified Library for Parameter-Efficient and Modular Transfer Learning", arXiv:2311.11077 , URL https://arxiv.org/abs/2311.11077

[17] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, Luke Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs", arXiv:2305.14314, URL https://arxiv.org/abs/2305.14314

[18] URL https://en.wikipedia.org/wiki/Tf%E2%80%93idf

[19] URL https://en.wikipedia.org/wiki/Latent_semantic_analysis

[20] URL https://en.wikipedia.org/wiki/Okapi_BM25

[21] URL https://docs.llamaindex.ai/en/stable/examples/query_transformations/SimpleIndexDemo-multistep/

[22] URL https://docs.llamaindex.ai/en/stable/api_reference/retrievers/query_fusion/

[23] URL https://docs.llamaindex.ai/en/stable/examples/query_transformations/HyDEQueryTransformDemo/

[24] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter", arXiv:1910.01108 , URL https://arxiv.org/abs/1910.01108

[25] S. Wild, S. Parlar, T. Hanne and R. Dornberger, "Naïve Bayes and Named Entity Recognition for Requirements Mining in Job Postings," 2021 3rd International Conference on Natural Language Processing (ICNLP), Beijing, China, 2021, pp. 155-161, doi: 10.1109/ICNLP52887.2021.00032.

[26] URL https://www.restack.io/p/natural-language-processing-knowledge-xgboost-nlp-cat-ai

[27] URL https://www.kaggle.com/code/bavalpreet26/ner-using-crf

[28] URL https://medium.com/@mail4sameera/named-entity-recognition-ner-in-natural-language-processing-8c3a357edb7f

[29] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, “BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805 , URL https://arxiv.org/abs/1810.04805

[30] URL https://spacy.io/universe/project/video-spacys-ner-model

[31] URL https://docs.llamaindex.ai/en/stable/examples/evaluation/relevancy_eval/

[32] URL https://docs.llamaindex.ai/en/stable/examples/evaluation/faithfulness_eval/

[33] URL https://spacy.io/models/en

[34] URL https://paperswithcode.com/dataset/ontonotes-5-0

[35] URL https://docs.llamaindex.ai/en/stable/api_reference/node_parsers/sentence_splitter/

[36] URL https://huggingface.co/BAAI/bge-small-en-v1.5

[37] URL https://medium.com/@Jawabreh0/inverted-file-indexing-ivf-in-faiss-a-comprehensive-guide-c183fe979d20

[38] URL https://docs.llamaindex.ai/en/stable/module_guides/indexing/vector_store_index/

[39] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, “Distilling the Knowledge in a Neural Network”, arXiv:1503.02531, URL https://arxiv.org/abs/1503.02531

[40] URl https://huggingface.co/HuggingFaceH4/zephyr-7b-alpha

[41] URL https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, “Attention is All You Need”, arXiv:1706.03762 , URL https://arxiv.org/abs/1706.03762

[43] URL https://huggingface.co/learn/nlp-course/en/chapter6/5

[44] URL https://paperswithcode.com/method/residual-connection

[45] URL https://huggingface.co/transformers/v3.5.1/model_doc/gpt.html

[46] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, Guoyin Wang, "Instruction Tuning for Large Language Models: A Survey", arXiv:2308.10792 , URL https://arxiv.org/abs/2308.10792

[47] URL https://huggingface.co/blog/rlhf

[48] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov, "Proximal Policy Optimization Algorithms", arXiv: 1707.06347, URL https://arxiv.org/abs/1707.06347

[49] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, Chelsea Finn, "Direct Preference Optimization: Your Language Model is Secretly a Reward Model", arXiv:2305.18290 , URL https://arxiv.org/abs/2305.18290

[50] URL https://cloud.google.com/discover/what-are-ai-hallucinations?hl=en

[51] URL https://pyterrier.readthedocs.io/en/latest/neural.html

[52] URL https://python.langchain.com/docs/integrations/document_transformers/cross_encoder_reranker/

[53] URL https://spacy.io/

[54] URL https://www.llamaindex.ai/

[55] URL https://huggingface.co/

[56] URL https://docs.llamaindex.ai/en/stable/examples/llm/nvidia/

[57] URL https://docs.ragas.io/en/latest/howtos/integrations/_llamaindex/#load-the-documents

[58] URL https://www.python.org/

[59] Department of Justice, URL https://doj.gov.in/

[60] Legislative Department of India, URL https://legislative.gov.in/

[61] IndiaFilings, URL https://www.indiafilings.com/

[62] LegalserviceIndia, URL https://www.legalserviceindia.com/

[63] iPLeaders Blog, URL https://blog.ipleaders.in/

Appendices

Chapter 1 - Introduction

[1.A] Platform and softwares used for the project 'LegalRAG':

1. Spacy [53] for Named Entity Recognition
2. LlamaIndex [54] for RAG
3. HuggingFace [55] and Nvidia cloud[56] for the LLMs and the embedding models
4. RAGAS[57] and Llamaindex used for Evaluation dataset creation
5. Python [58] for coding

[1.B] Data Collection:

Data was collected from the following locations regarding the four legal acts in consideration (copyright, inheritance, consumer protection and divorce):

1. Department of Justice of India [59]
2. Legislative Department of India [60]
3. Indiafilings blog website [61]
4. LegalServiceIndia blog website [62]
5. iPLeaders blog articles (from LawSikho) [63]

Data for each of the four acts collected from these multiple locations were combined to form a single document. Ensuring there is one document per act. This is not a mandatory step as the data reading softwares and platforms are capable of handling multiple documents per act, but this was done as a step to ensure there is manual care, trust and oversight on the data being collected.

[1.C] Data, Code, and results

1. All the data collected, codes written, results and reports of LegalRAG can be found in this Google drive: https://drive.google.com/drive/folders/1ofAIXxYwBtZYTs_7kIRJRGcAar8cAuE6?usp=sharing
2. Codes can also be found in this github repo: https://github.com/sachinhebbar891/LegalRAG

Chapter 2 - Literature Review

None

Chapter 3 - Methodology

None

Chapter 4 - Data Preprocessing

[4.A] Here is an example of the original data with PII information such as the name of the person involved in divorce

Figure 18: Data with unmasked PII

```
text = """

The Shah Bano case became a landmark in Indian legal history, setting a precedent for maintenance rights for Muslim women.
Shah Bano, a Muslim woman, was divorced by her husband and sought maintenance under Section 125 of the Criminal Procedure Code.
The Supreme Court initially ruled in her favor, but due to political pressure, the government passed the Muslim Women (Protection of Rights on Divorce) Act in 1986,
limiting the maintenance duration for Muslim women. This case highlighted the need for a uniform civil code to ensure equal rights for women across all religions.

"""
```

Note that the name shown in the above text is a made up name and not the real name of the person involved in those divorce cases.

[4.B] Here is an example after masking the original data that had PII information (divorcee's name) for the text of 4.A

Figure 19: Data with masked PII

```
'\n\nthe **** case became a landmark in indian legal history, setting a precedent for maintenance rights for muslim women.\n****, a muslim woman, was divorced by her husband and sough
t maintenance under section 125 of the criminal procedure code. \nthe supreme court initially ruled in her favor, but due to political pressure, the government passed the muslim women
(protection of rights on divorce) act in 1986, \nlimiting the maintenance duration for muslim women. this case highlighted the need for a uniform civil code to ensure equal rights for
women across all religions.\n\n'
```

[4.C] Chunking

Below code snippet shows a SentenceSplitter chunking with a chunk size of 100 and chunk overlap of 50

Figure 20: SentenceSplitter code snippet

```
text_splitter = SentenceSplitter(
   separator=" ",
   chunk_size=100,
   chunk_overlap=50

)
```

[4.D] Embedding

Below code snippet shows loading the embedding model bge-small-en-v1.5 from HuggingFace.

Figure 21: Embedding model

```
embed_model = HuggingFaceEmbedding(model_name="BAAI/bge-small-en-v1.5")
```

```
modules.json: 100% 349/349 [00:00<00:00, 30.4kB/s]
config_sentence_transformers.json: 100% 124/124 [00:00<00:00, 11.4kB/s]
README.md: 100% 94.8k/94.8k [00:00<00:00, 2.17MB/s]
sentence_bert_config.json: 100% 52.0/52.0 [00:00<00:00, 4.81kB/s]
config.json: 100% 743/743 [00:00<00:00, 72.6kB/s]
model.safetensors: 100% 133M/133M [00:00<00:00, 230MB/s]
tokenizer_config.json: 100% 366/366 [00:00<00:00, 32.9kB/s]
vocab.txt: 100% 232k/232k [00:00<00:00, 1.82MB/s]
tokenizer.json: 100% 711k/711k [00:00<00:00, 3.47MB/s]
special_tokens_map.json: 100% 125/125 [00:00<00:00, 11.0kB/s]
1_Pooling/config.json: 100% 190/190 [00:00<00:00, 15.5kB/s]
```

[4.E] Storing data in VectorStore

Below code snippet shows storing data in Chromadb (an open source vector store used in LegalRAG)

Figure 22: Vector store

```
chroma_client = chromadb.EphemeralClient()
chroma_collection = chroma_client.create_collection("quickstart")

vector_store = ChromaVectorStore(chroma_collection=chroma_collection)
storage_context = StorageContext.from_defaults(vector_store=vector_store)
index = VectorStoreIndex(nodes, storage_context=storage_context, embed_model=embed_model)
```

```
# save the data in a disk
db2 = chromadb.PersistentClient(path="./chroma_db")
# saves the data as chroma.sqlite3
```

```
# loading the data from disk
chroma_collection = db2.get_or_create_collection("quickstart")
vector_store = ChromaVectorStore(chroma_collection=chroma_collection)
index = VectorStoreIndex.from_vector_store(
    vector_store,
    embed_model=embed_model,
)
```

Chapter 5 - Intent Classification

[5.A] Training and test set creation for intent classification

A train dataset was created manually ensuring 40 examples for each intent. Care was taken to ensure the examples for each intent was as diverse as possible to ensure the model is better able to predict the correct intent.

The following table shows 3 records from the train dataset for each of the intent:

Table 11: Intent classification training sample

Query	Intent
What are the grounds for divorce?	Divorce
How long does the divorce process take?	Divorce
Can I file for divorce online?	Divorce
What is the time limit for filing a consumer complaint?	Consumer protection
How to report misleading advertisements?	Consumer protection

What is the process for getting a refund for a faulty product?	Consumer protection
Who inherits my father's wealth?	Inheritance
How to resolve inheritance disputes?	Inheritance
What is the process of writing a will?	Inheritance
What are my rights as a copyright owner?	Copyright
What is copyright infringement?	Copyright
How to register a copyright?	Copyright
Where is the closest hospital?	Other
How to get rich?	Other
What is the best way to learn a new language?	Other

A test set was created with 10 records for each of the intent, ensuring that these records are not the same as the ones in the training dataset

[5.B] distilBERT

DistilBERT was used to create the intent recognition model. The pre-trained version of distibert was taken from HuggingFace: distilbert-base-uncased

A code snippet on loading distilBERT

Figure 23: DistilBERT

```
checkpoint = "distilbert-base-uncased"
classifier = AutoModelForSequenceClassification.from_pretrained(checkpoint, num_labels=5) # Get the classifier
tokenizer = AutoTokenizer.from_pretrained(checkpoint)
```

```
config.json: 100% 483/483 [00:00<00:00, 30.5kB/s]
model.safetensors: 100% 268M/268M [00:01<00:00, 206MB/s]
Some weights of DistilBertForSequenceClassification were not initialized from the model checkpoint at distilbert-base-uncased
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
tokenizer_config.json: 100% 48.0/48.0 [00:00<00:00, 1.46kB/s]
vocab.txt: 100% 232k/232k [00:00<00:00, 4.88MB/s]
tokenizer.json: 100% 466k/466k [00:00<00:00, 11.4MB/s]
```

[5.C] Training results of distilBERT

Figure 24: DistilBERT training results

[125/125 04:25, Epoch 5/5]

Epoch	Training Loss	Validation Loss	Accuracy	Precision	Recall	F1
1	No log	0.710698	1.000000	1.000000	1.000000	1.000000
2	No log	0.175783	0.980000	0.981818	0.980000	0.979950
3	No log	0.095075	0.980000	0.981818	0.980000	0.979950
4	No log	0.088341	0.980000	0.981818	0.980000	0.979950
5	No log	0.081049	0.960000	0.966667	0.960000	0.960766

Testing results of distilBERT are shown in the chapter on Results.

[5.D] Intent classification in action on unseen user queries

The following question “Hope I get rich someday” has been correctly classified as ‘Other’ intent

Figure 25: Other intent query

```
# Example input text
new_text = "Hope I get rich someday"

# Tokenize the new text (same as during training)
inputs = tokenizer(new_text, return_tensors="pt", padding='max_length', truncation=True, max_length=128)

# Convert tokenized input into a Dataset
predict_dataset = Dataset.from_dict({
    'input_ids': inputs['input_ids'],
    'attention_mask': inputs['attention_mask']
})

# Get predictions from the trainer
predictions = trainer.predict(predict_dataset)

# Extract the logits
logits = predictions.predictions

# Get the index of the highest logit (the predicted class)
predicted_label = torch.argmax(torch.tensor(logits), dim=1).item()

print(f"Predicted intent: {intent_label_dict[predicted_label]}")
```

```
Predicted intent: Other
```

The following question: “I just want divorce!” has been correctly classified as ‘Divorce’ intent:

Figure 26: Divorce intent query

```
# Example input text
new_text = "I just want divorce!"

# Tokenize the new text (same as during training)
inputs = tokenizer(new_text, return_tensors="pt", padding='max_length', truncation=True, max_length=128)

# Convert tokenized input into a Dataset
predict_dataset = Dataset.from_dict({
    'input_ids': inputs['input_ids'],
    'attention_mask': inputs['attention_mask']
})

# Get predictions from the trainer
predictions = trainer.predict(predict_dataset)

# Extract the logits
logits = predictions.predictions

# Get the index of the highest logit (the predicted class)
predicted_label = torch.argmax(torch.tensor(logits), dim=1).item()

print(f"Predicted intent: {intent_label_dict[predicted_label]}")
```

```
Predicted intent: Divorce
```

The following question: "How can I protect my rights as a consumer?” has been correctly classified as ‘Consumer protection’ intent.

Figure 27: Consumer protection intent query

```
query = "How can I protect my rights as a consumer?"
intent_inference(query, model, tokenizer, intent_label_dict)
```

```
Device set to use cuda:0
/usr/local/lib/python3.11/dist-packages/transformers/pipelines
  warnings.warn(
'Consumer protection'
```

The following question: “Who is a legal heir?” has been correctly classified as ‘Inheritance’ intent.

Figure 28: Inheritance intent query

```
user_query = "who is a legal heir?"
intent = intent_inference(user_query, mode
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent,
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Inheritance
```

The following question: “What is copyright infringement?” has been correctly classified as ‘Copyright’ intent.

Figure 29: Copyright intent query

```
user_query = "what is copyright infringement?"
intent = intent_inference(user_query, model, toke
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_qu
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Copyright
```

Chapter 6 - Large Language Model

None

Chapter 7 - Retrieval Augmented Generation

[7.A] Code snippet for simple retrieval

Figure 30: Simple retrieval code snippet

```python
def rag(user_query, index):

  # simple retrieval that uses cosine similarity and gets top 5 most similar records
  query_engine = index.as_query_engine(similarity_top_k = 5)


  response = query_engine.query(user_query)

  return response, response.source_nodes
```

[7.B] RAG results for simple retrieval

For Copyright

Figure 31: Response for a copyright related query using simple retrieval

```python
user_query = "what is copyright infringement?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Copyright
Response:
'A criminal offense punishable under a specific section of the Copyright Act, with a minimum punishment of imprisonment for six months and a fine of Rs.50,000/-, resulting from the un
authorized use or reproduction of a copyrighted work.'
```

Figure 32: Simple Retrieval for copyright related query

```python
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
Remedy for copyright infringement
Copyright infringement of any work is a criminal offense punishable under Section 63 of the Copyright Act. The minimum punishment for infringement of a copyright is imprisonment for si
---------------
Copyright registration is one of the key types of intellectual property protection and allows for the protection of literary, dramatic, musical and artistic works.
---------------
In this article we look at the basics of copyright registration in India:
Copyright
Copyright is a right to ownership and enjoyment given by the law to creators of literary, dramatic, musical, artistic works and producers of cinematograph films and sound recordings.
---------------
It is a bundle of rights comprising of rights to reproduction, communication to the public, adaptation and translation of the work. Copyright ensures certain minimum safeguards of the
What can be copyrighted?
---------------
Also, in case a copyright infringement has happened or happening or likely to happen, any police officer, not below the rank of a sub inspector, may, if he is satisfied, seize without
---------------
```

For Divorce

Figure 33: Response for a divorce related query using simple retrieval

```python
user_query = "How can I get divorce?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Divorce
Response:
'A Christian couple can get a divorce with mutual consent (no-fault divorce or mutual divorce), or either spouse may file for divorce without the consent of the other (fault divorce)
as per Indian divorce act.'
```

Figure 34: Simple Retrieval for divorce related query

```
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
* The couples should be separated for over a year
* The couple should able to prove that they have not been able to live together
* Matters of children’s custody,
---------------
Types of Divorce Petitions
A Christian couple can get a divorce with mutual consent (no-fault divorce or mutual divorce), or either spouse may file for divorce without the consent of the other (fault divorce) as
---------------
The wife can file such petition under any of the following circumstance:
* If her husband has exchanged his profession of Christianity for the profession of some other religion
* If the husband went through a form of marriage with another woman
* If her husband has been guilty of incestuous adultery since the solemnization
---------------
Divorce is followed by granting alimony, child custody, and child visitation, distribution of property and distribution of debts. Before opting for a divorce, the Christian couple shou
---------------
The result of the enquiry and the additional evidence will be certified to the High Court by the District Judge, and the High Court will make an order confirming the decree for dissolu
---------------
```

For Consumer Protection

Figure 35: Response for a consumer protection related query using simple retrieval

```
user_query = "How to file a complaint for damaged goods?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Consumer protection
Response:
'A consumer complaint relating to damaged goods must be filed in writing with a District Forum by the consumer along with the fee.'
```

Figure 36: Simple Retrieval for consumer protection related query

```
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
Procedure to File Consumer Case
Any consumer complaint relating to a good or service must be filed in writing with a District Forum by the consumer along with the fee. On receipt of a complaint, the District Forum ma
---------------
How and When to Complain
Under the Consumer Protection Act, the customer can raise a complaint in writing if:
* Adoption of any unfair trade practise or a restrictive trade practice by any trader or service provider;
* The goods bought by him or agreed to be bought by him suffer from one or more defects;
---------------
as the case may be, has charged for the goods or for the services mentioned in the complaint, a price in excess of the price:
   * Fixed by or under any law for the time being in force;
   * Displayed on the goods or any package containing such goods;
---------------
* The services hired or availed of or agreed to be hired or availed of by him suffer from a deficiency in any respect;
* Trader or the service provider, as the case may be, has charged for the goods or for the services mentioned in the complaint,
---------------
On the other hand, a copy of the complaint will reach the opposite party for the approval to be made within 45 days.
The effort is taken to hear all consumer cases as expeditiously as possible.
---------------
```

For Inheritance

Figure 37: Response for a inheritance related query using simple retrieval

```
user_query = "Who is a legal heir?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Inheritance
Response:
'An individual who takes the place of the property of his/her ancestor, either by law or by a will.'
```

Figure 38: Simple Retrieval for inheritance related query

```
for node in source_nodes:
  print(node.text)
  print("--------------")
```

```
Therefore a legal heir is an individual who takes the place of the property of his/her ancestor, either by law or by a will.
Importance of heir
It is essential to identify a legal heir for every person owning property; they are the successors for property claims and insurance coverage.
--------------
Legal heir in India?
According to India laws, a person who is determined to succeed to the estate of an ancestor who has died without making a will or mentioning a legal heir.
--------------
Legal heir under Hindu law
Under the Hindu Succession (Amendment) Act, 2005, the following can be the legal heir of a person: Class I
* Wife (Widow)
* Mother
* Son
* Daughter
* Deceased son's daughter
* Deceased daughter's
--------------
legal heirs of a Parsi person are:
* Father
* Mother
* Full brother
* Full sister
* Paternal grandparents
* Maternal grandparents
* Children of maternal grandparents and their lineal descendants
* Children of paternal grandparents and their lineal descendants
* Parents of paternal grandparents
* Parents
--------------
```

[7.C] Code snippet for Query fusion

Figure 39: Query fusion code snippet

```python
def rag(user_query, index):

  retriever = QueryFusionRetriever(
    [index.as_retriever()],
    similarity_top_k=3,
    num_queries=4,  # set this to 1 to disable query generation
    use_async=True,
    verbose=True,
    )

  query_engine = RetrieverQueryEngine.from_args(retriever)

  response = query_engine.query(user_query)

  return response, response.source_nodes
```

[7.D] RAG results for Query fusion

For Divorce

Figure 40: Generated queries for divorce related query

```python
user_query = "How can I get alimony in divorce?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
```

```
Device set to use cuda:0
Intent of the query: Divorce
Generated queries:
Here are three search queries related to the input query:
How can I get alimony in divorce?
What are the eligibility criteria for alimony in a divorce settlement?
How do I calculate the amount of alimony I'm entitled to in a divorce?
```

Figure 41: Response for divorce related query using query fusion

```python
print("Response:")
response.response
```

```
Response:
'In India, divorce after a decade of marriage entitles the spouse to life-long alimony. The amount of alimony is determined by considering factors such as the age of the person entitl
ed to receive alimony, their economic condition or earnings potential, and their health. Additionally, alimony can also be claimed by dependent children and indigent parents. The spou
se seeking alimony must demonstrate that the other spouse has sufficient means to provide for their support.'
```

Figure 42: Query fusion retrieval for divorce related query

```
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
are within the prohibited degrees of consanguinity or affinity
* In case of either party was a lunatic or idiot at the time of the marriage
* In case the former husband or wife of either party was living at the time of the solemnization,
---------------
* The couples should be separated for over a year
* The couple should able to prove that they have not been able to live together
* Matters of children's custody,
---------------
Types of Divorce Petitions
A Christian couple can get a divorce with mutual consent (no-fault divorce or mutual divorce), or either spouse may file for divorce without the consent of the other (fault divorce) as
---------------
```

For Consumer Protection

Figure 43: Generated queries for consumer protection related query

```
user_query = "How to file a complaint for damaged goods?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
```

```
Device set to use cuda:0
Intent of the query: Consumer protection
Generated queries:
Here are three search queries related to the input query:
How to file a complaint for damaged goods?
1. "Filing a complaint with a shipping company for damaged goods"
2. "What are the steps to file a complaint for damaged products"
3. "How to report damaged goods to the manufacturer for a refund"
```

Figure 44: Response for consumer protection related query using query fusion

```
print("Response:")
response.response
```

```
Response:
'A consumer complaint in writing must be filed with a District Forum, along with the fee, if the goods bought by the consumer suffer from one or more defects.'
```

Figure 45: Query fusion retrieval for consumer protection related query

```
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
How and When to Complain
Under the Consumer Protection Act, the customer can raise a complaint in writing if:
* Adoption of any unfair trade practise or a restrictive trade practice by any trader or service provider;
* The goods bought by him or agreed to be bought by him suffer from one or more defects;
---------------
Procedure to File Consumer Case
Any consumer complaint relating to a good or service must be filed in writing with a District Forum by the consumer along with the fee. On receipt of a complaint, the District Forum ma
---------------
```

For Inheritance

Figure 46: Generated queries for inheritance related query

```
user_query = "Who is a legal heir?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
```

```
Device set to use cuda:0
Intent of the query: Inheritance
Generated queries:
Here are three search queries related to the input query "Who is a legal heir?":
Who is a legal heir in India?
What are the rules for determining a legal heir?
How do you prove legal heirship in a court of law?
```

Figure 47: Response for inheritance related query using query fusion

```
print("Response:")
response.response
```

```
Response:
'An individual who takes the place of the property of his/her ancestor, either by law or by a will.'
```

Figure 48: Query fusion retrieval for inheritance related query

```
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
Legal heir in India?
According to India laws, a person who is determined to succeed to the estate of an ancestor who has died without making a will or mentioning a legal heir.
---------------
Therefore a legal heir is an individual who takes the place of the property of his/her ancestor, either by law or by a will.
Importance of heir
It is essential to identify a legal heir for every person owning property; they are the successors for property claims and insurance coverage.
---------------
Legal heir under Hindu law
Under the Hindu Succession (Amendment) Act, 2005, the following can be the legal heir of a person: Class I
* Wife (Widow)
* Mother
* Son
* Daughter
* Deceased son's daughter
* Deceased daughter's
---------------
```

For Copyright

Figure 49: Generated queries for copyright related query

```
user_query = "what is copyright infringement?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
```

```
Device set to use cuda:0
Intent of the query: Copyright
Generated queries:
Here are three search queries related to the input query "what is copyright infringement?":
What is the legal definition of copyright infringement?
How to identify and prevent copyright infringement in business?
What are the consequences of copyright infringement and how to avoid them?
```

Figure 50: Response for copyright related query using query fusion

```
print("Response:")
response.response
```

```
Response:
'Copyright infringement of any work is a criminal offense punishable under Section 63 of the Copyright Act.'
```

Figure 51: Query fusion retrieval for copyright related query

```
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
Remedy for copyright infringement
Copyright infringement of any work is a criminal offense punishable under Section 63 of the Copyright Act. The minimum punishment for infringement of a copyright is imprisonment for si
---------------
Copyright registration is one of the key types of intellectual property protection and allows for the protection of literary, dramatic, musical and artistic works.
---------------
Copyright registration
A limited amount of copyright protection comes into existence as soon as a work is created and no formality is required to be completed for acquiring copyright.
---------------
```

[7.E] Code snippet for HyDE

Figure 52: HyDE code snippet

```
# Function to generate HyDE document and perform retrieval and response generation from the LLM
def rag(user_query, index):

  # simple retrieval that uses cosine similarity and gets top 5 most similar records
  query_engine = index.as_query_engine(similarity_top_k = 5)
  # Creating HyDE retrieval
  hyde = HyDEQueryTransform(include_original=True)
  hyde_query_engine = TransformQueryEngine(query_engine, hyde)

  # generating HyDE document
  query_bundle = hyde(user_query)
  hyde_doc = query_bundle.embedding_strs[0]

  # using HyDE retrieval to get context and use the context to generate a response from the LLM
  response = hyde_query_engine.query(user_query)

  return response, response.source_nodes, hyde_doc
```

[7.F] RAG results for HyDE

For Copyright

Figure 53: Response for a copyright related query using HyDE

```python
user_query = "what is copyright infringement?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes, hyde_doc = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Copyright
Response:
'A criminal offense punishable under Section 63 of the Copyright Act, with a minimum punishment of imprisonment for six months and a minimum fine of Rs.50,000/-
```

Figure 54: HyDE response for copyright related query

```python
print("HyDE response:")
print(hyde_doc)
```

```
HyDE response:
Here is a passage that answers the question:

Copyright infringement is the unauthorized use, reproduction, or distribution of someone else's original work, such as a book, song, movie, or piece of art, without their permission or

In the United States, copyright law is governed by the Copyright Act of 1976, which grants creators of original works the exclusive right to reproduce, distribute, and display their wo

Examples of copyright infringement include:

* Downloading or sharing copyrighted music or movies without permission
* Copying or distributing copyrighted software or video games without permission
* Using copyrighted images or artwork without permission
* Uploading copyrighted content to a website or social media platform without permission
* Creating a derivative work, such as a parody or remix, without permission

Copyright infringement is a serious issue, as it can cause significant financial losses and harm to creators and artists. It is important for individuals and businesses to understand t
```

Figure 55: HyDE retrieval for copyright related query

```python
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
Remedy for copyright infringement
Copyright infringement of any work is a criminal offense punishable under Section 63 of the Copyright Act. The minimum punishment for infringement of a copyright is imprisonment for si
---------------
In this article we look at the basics of copyright registration in India:
Copyright
Copyright is a right to ownership and enjoyment given by the law to creators of literary, dramatic, musical, artistic works and producers of cinematograph films and sound recordings.
---------------
It is a bundle of rights comprising of rights to reproduction, communication to the public, adaptation and translation of the work. Copyright ensures certain minimum safeguards of the
What can be copyrighted?
---------------
Also, in case a copyright infringement has happened or happening or likely to happen, any police officer, not below the rank of a sub inspector, may, if he is satisfied, seize without
---------------
Copyright registration is one of the key types of intellectual property protection and allows for the protection of literary, dramatic, musical and artistic works.
---------------
```

For Divorce

Figure 56: Response for a divorce related query using HyDE

```python
user_query = "How can I get alimony after divorce?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes, hyde_doc = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Divorce
Response:
'In a divorce after a decade of marriage, the spouse is entitled to life-long alimony. The court will consider several factors, including the age of the person receiving alimony, thei
r economic condition, and the health of the spouse. The court will also take into account the earning potential of the husband, his ability to regenerate his fortune, and his liabilit
ies.'
```

Figure 57: HyDE response for divorce related query

```
print("HyDE response:")
print(hyde_doc)
```

```
HyDE response:
When it comes to getting alimony after a divorce, there are several key factors to consider. First and foremost, alimony is typically awarded to the spouse who is in a more financially

To be eligible for alimony, the dependent spouse must demonstrate a need for financial support, which can be established through documentation of their income, expenses, and assets. Th

In determining the amount of alimony, the court will consider several factors, including the length of the marriage, the age and health of both spouses, the income and earning capacity

In addition to these factors, the court may also consider the supporting spouse's ability to pay alimony, including their income, expenses, and assets. The court may also consider any

If you are seeking alimony after a divorce, it is essential to work with an experienced family law attorney who can help you navigate the complex legal process. Your attorney can help
```

Figure 58: HyDE retrieval for divorce related query

```
for node in source_nodes:
  print(node.text)
  print("---------------")
```

```
Divorce after a decade of marriage entitles the spouse to life-long alimony. The other important factors that need to be considered are given here:
* Age of the person who is entitled to receive the alimony
* The economic condition or the earnings potential of the person who is entitled to pay alimony
* The health of spouse,
---------------
Divorce after a decade of marriage entitles the spouse to life-long alimony.
---------------
While deciding that the alimony is to be paid, the courts will take into account the earning potential of the husband, his ability to regenerate his fortune (the property is given to t
---------------
This will include, spouse, dependent children and even indigent parents.
Computing amount of alimony
The alimony claim of either spouse (though, in the vast majority of cases, it is the wife) depends on whether the husband has sufficient means.
---------------
Note: In some cases, the court can make an order to the husband for payment to the wife of monthly or weekly sums for her maintenance and support
Factors that influence the amount of alimony
The alimony amount depends upon the length of marriage in a contested divorce,
---------------
```

For Consumer Protection

Figure 59: Response for a consumer protection related query using HyDE

```
user_query = "How to file a complaint for damaged goods?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes, hyde_doc = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Consumer protection
Response:
'A consumer complaint relating to a good must be filed in writing with a District Forum by the consumer along with the fee.'
```

Figure 60: HyDE response for consumer protection related query

```
print("HyDE response:")
print(hyde_doc)
```

```
HyDE response:
Filing a complaint for damaged goods can be a straightforward process if you follow the right steps. First, it's essential to document the damage by taking clear photos or videos of th

If you purchased the item online, you may also need to contact the online retailer's customer service department. They will likely have a specific process for handling damaged goods ar

In addition to contacting the seller or manufacturer, you may also want to file a complaint with the relevant consumer protection agency or regulatory body. This can help ensure that y

Finally, keep a record of all correspondence and interactions with the seller, manufacturer, and any regulatory agencies. This will help you track the progress of your complaint and er
```

Figure 61: HyDE retrieval for consumer protection related query

```
for node in source_nodes:
  print(node.text)
  print("--------------")
```

```
Procedure to File Consumer Case
Any consumer complaint relating to a good or service must be filed in writing with a District Forum by the consumer along with the fee. On receipt of a complaint, the District Forum ma
--------------
How and When to Complain
Under the Consumer Protection Act, the customer can raise a complaint in writing if:
* Adoption of any unfair trade practise or a restrictive trade practice by any trader or service provider;
* The goods bought by him or agreed to be bought by him suffer from one or more defects;
--------------
as the case may be, has charged for the goods or for the services mentioned in the complaint, a price in excess of the price:
   * Fixed by or under any law for the time being in force;
   * Displayed on the goods or any package containing such goods;
--------------
On the other hand, a copy of the complaint will reach the opposite party for the approval to be made within 45 days.
The effort is taken to hear all consumer cases as expeditiously as possible.
--------------
The effort is taken to hear all consumer cases as expeditiously as possible. Further, it is endeavoured to decide the complaint within a period of three months from the date of receipt
--------------
```

For Inheritance

Figure 62: Response for a inheritance related query using HyDE

```
user_query = "Who is a legal heir?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes, hyde_doc = legalRAG(intent, user_query, indexes)
print("Response:")
response.response
```

```
Device set to use cuda:0
Intent of the query: Inheritance
Response:
'An individual who takes the place of the property of his/her ancestor, either by law or by a will.'
```

Figure 63: HyDE response for inheritance related query

```
print("HyDE response:")
print(hyde_doc)
```

```
HyDE response:
Here is a passage that answers the question:

A legal heir is an individual who inherits property, assets, or rights from a deceased person, typically in accordance with the laws of the country or jurisdiction. In general, a legal
```

Figure 64: HyDE retrieval for inheritance related query

```
for node in source_nodes:
  print(node.text)
  print("--------------")
```

```
Therefore a legal heir is an individual who takes the place of the property of his/her ancestor, either by law or by a will.
Importance of heir
It is essential to identify a legal heir for every person owning property; they are the successors for property claims and insurance coverage.
--------------
Legal heir in India?
According to India laws, a person who is determined to succeed to the estate of an ancestor who has died without making a will or mentioning a legal heir.
--------------
Legal heir under Hindu law
Under the Hindu Succession (Amendment) Act, 2005, the following can be the legal heir of a person: Class I
* Wife (Widow)
* Mother
* Son
* Daughter
* Deceased son's daughter
* Deceased daughter's
--------------
legal heirs of a Parsi person are:
* Father
* Mother
* Full brother
* Full sister
* Paternal grandparents
* Maternal grandparents
* Children of maternal grandparents and their lineal descendants
* Children of paternal grandparents and their lineal descendants
* Parents of paternal grandparents
* Parents
```

[7.G] Results for Other intent (for all 3 retrieval mechanisms)

Figure 65: Response for Other queries

```
user_query = "Where to go for vacation?"
intent = intent_inference(user_query, model, tokenizer, intent_label_dict)
print("Intent of the query:", intent)
response, source_nodes = legalRAG(intent, user_query, indexes)
print("Response:")
print(response)
```

```
Device set to use cuda:0
Intent of the query: Other
Response:
Please ask questions related to Divorce act, inheritance act, consumer protection act, or copyright act
```

Chapter 8 - Evaluation metrics

[8.A] Code snippet showing the evaluation metrics for the RAG system

Figure 66: Evaluation metrics code snippets

```
relevancy_evaluator = RelevancyEvaluator(llm=evaluation_llm)
faithfulness_evaluator = FaithfulnessEvaluator(llm=evaluation_llm)
similarity_evaluator = SemanticSimilarityEvaluator()
correctness_evaluator = CorrectnessEvaluator(llm=evaluation_llm)
```

[8.B] Code snippet where scores are calculated (shown only for copyright, but it is similar for others)

Figure 67: Evaluation scores calculation code snippet

```
queries = df_copyright1["Query"].tolist()
reference_responses = df_copyright1['Reference'].tolist()
responses = []
relevancy_scores = []
faithfulness_scores = []
semantic_similarity_scores = []
correctness_scores = []

for index in range(0, len(queries)):
  response = query_engine_copyright1.query(queries[index])
  relevancy_score = relevancy_evaluator.evaluate_response(queries[index], response)
  faithfulness_score = faithfulness_evaluator.evaluate_response(response = response)

  similarity_score = similarity_evaluator.evaluate(response = response.response, reference = reference_responses[index])

  correctness_score = correctness_evaluator.evaluate(queries[index], response.response, reference_responses[index])

  relevancy_scores.append(relevancy_score.passing)
  faithfulness_scores.append(faithfulness_score.passing)
  semantic_similarity_scores.append(similarity_score.score)
  correctness_scores.append(correctness_score.score)
  responses.append(response.response)
```

Chapter 9 - Results

[9.A] Evaluation dataset creation

The evaluation dataset is created partially with the help of Llama3-8b-instruct. Questions and contexts are prepared by the LLM. These are manually verified and responses are added

Figure 68: Evaluation dataset creation code snippet

```
qa_dataset = generate_question_context_pairs(
    nodes1, llm=llm, num_questions_per_chunk=2
)
```

```
100%|██████████| 18/18 [00:14<00:00,  1.28it/s]
```

```
qa_dataset2 = generate_question_context_pairs(
    nodes2, llm=llm, num_questions_per_chunk=2
)
```

```
100%|██████████| 72/72 [00:56<00:00,  1.27it/s]
```

```
qa_dataset3 = generate_question_context_pairs(
    nodes3, llm=llm, num_questions_per_chunk=2
)
```

```
100%|██████████| 16/16 [00:13<00:00,  1.21it/s]
```

```
qa_dataset4 = generate_question_context_pairs(
    nodes4, llm=llm, num_questions_per_chunk=2
)
```

```
100%|██████████| 47/47 [00:35<00:00,  1.31it/s]
```

```
qa_dataset.save_json("/content/" + "copyright_eval_dataset.json")
qa_dataset2.save_json("/content/" + "divorce_eval_dataset.json")
qa_dataset3.save_json("/content/" + "cp_eval_dataset.json")
qa_dataset4.save_json("/content/" + "inheritance_eval_dataset.json")
```

[9.B] Screenshots of the Evaluation dataset

Evaluation dataset of copyright legal act with questions, context and response

Figure 69: Evaluation dataset for copyright

Query	Context	Reference
What is the primary purpose of copyright, according to the given context?	It is a bundle of rights comprising of rights to reproduction, communication to the public, adaptation and translation of the work. Copyright ensures certain minimum safeguards of the rights of ownership and enjoyment of the authors over their creations, thereby protecting and rewarding creativity.\r\nWhat can be copyrighted?	The primary purpose of copyright is to ensure certain minimum safeguards of the rights of ownership and enjoyment of the authors over their creations, thereby protecting and rewarding creativity.
What is the form that needs to be filled out to apply for copyright registration?	To apply for copyright, application for registration is to be made on Form IV in the requisite manner along with the applicable fee. Both published and unpublished works can be copyrighted. In case of published work, three copies of the published work has to be presented along with the application.	Form IV needs to be filled out to apply for copyright resgistration in the requisite manner along with the applicable fee.
What is the typical duration of copyright protection?	Copyright protection typically lasts for 60 years. In the case of original literary, dramatic, musical and artistic works the 60-year period is counted from the year following the death of the author.	Typical duration of copyright duration is 60 years.
What is the minimum punishment for copyright infringement under the Copyright Act?	Remedy for copyright infringement\r\nCopyright infringement of any work is a criminal offense punishable under Section 63 of the Copyright Act. The minimum punishment for infringement of a copyright is imprisonment for six months with a minimum fine of Rs.50,000/-.	The minimum punishment for infringement of a copyright is imprisonment for six months with a minimum fine of Rs 50,000/-
What is the minimum rank of a police officer who can seize copies of a work and plates used for making infringing copies without a warrant?	Also, in case a copyright infringement has happened or happening or likely to happen, any police officer, not below the rank of a sub inspector, may, if he is satisfied, seize without warrant, all copies of the work and all plates used for the purpose of making infringing copies of the work.	The minimum rank of a police officer who can seize copies of a work and plates used for making infringing copies without a warrant is the rank of sub-inspector

Evaluation dataset of divorce legal act with questions, context and response

Figure 70: Evaluation dataset for divorce

Query	Context	Reference
What type of divorce is considered by the courts when the couples agree to a divorce?	Divorce with Mutual Consent\r\nWhen the couples agree to a divorce, the courts will consider a divorce with mutual consent as per.	Divorce with Mutual consent
What is the minimum duration of separation required for a couple to file for divorce under Section 10A of the Indian Divorce Act, 1869?	Section 10A of Indian Divorce Act, 1869, requires the couple to be separated for at least two years, the couple only needs to provide that they have not been living as husband and wife during this period.	The minimum duration of sepration required for a couple to file divorce under Section 10A of the Indian Divorce Act is two years
What are the three aspects that couples need to reach a consensus on, according to the given context?	maintenance and property rights need to be agreed to mutually\r\nAlimony or Maintenance Issues\r\nThere are three aspects regarding which the couples have to reach a consensus. One is alimony or maintenance issues.	The three aspects that couples needto reach a consensus on is maintenance, property rights and alimony
What types of expenses will be considered when issuing a decree for divorce?	Custody of child, alimony to wife and litigation expenses will be considered on issuing a decree for divorce.	Expenses considered while issuing a decree of divorce are custody of child, alimony to wife and litigation expenses.
What is one circumstance that may favor the spouse receiving alimony?	the failing health or a medical condition of one of the spouses who is going to receive the alimony may act in favour of him or her.	Failing health or a medical condition of the spouse receiving the alimony may favor the spouse receiving alimony

Evaluation dataset of consumer protection legal act with questions, context and response

Figure 71: Evaluation dataset for consumer protection

Query	Context	Reference
What is the main law that provides protection to consumers in India, as mentioned in the context information?	\ufeffConsumer Protection Laws in India\r\nConsumer Protection Act is one of the main laws that provide protection to consumers in India. The Act was introduced in the year 1986 and then amended in the year 2002 through the Consumer Protection Amendment Act, 2002.	Consumer Protection Amendement Act, 2002 is the main law that provides protection to consumers in India.
What is the primary objective of the Consumer Protection Act, as mentioned in the article?	In this article, we look at the protection afforded to the consumers through the Act.\r\nObjective of the Consumer Protection Act\r\nThe main objective of the Consumer Protection Act is to provide better protection of consumers and establish a strong mechanism for the settlement of consumer disputes.	The primary objective of the Consumer Protection Act is to provide better protection of consumers and establish a strong mechanism for the settlement of consumer disputes.
What is the procedure to file a complaint in consumer court?	Procedure to File Consumer Case\r\nAny consumer complaint relating to a good or service must be filed in writing with a District Forum by the consumer along with the fee. On receipt of a complaint, the District Forum may reject or approve the complaint, usually within 21 days from the date of complaint.	Any consumer complaint relating to a good or service must be filed in writing with a District Forum by the consumer along with the fee. On receipt of a complaint, the District Forum may reject or approve the complaint, usually within 21 days from the date of complaint.
What is the possible outcome if the complainant fails to appear on the date of hearing before the District Forum?	Hence, if during the proceedings, the complainant fails to appear on the date of hearing before the District Forum, the District Forum may either dismiss the complaint about default or decide it on merits.	If the complainant fails to appear on the date of hearing before District Forum, the forum may either dismiss the complaint about default or decide it on merits.
What kind of goods and services does consumer protection account consider?	The Act covers all goods and services including banking, e-commerce, telecom, insurance, electricity, transportation in the private and public sector.\r\nHow and When to Complain\r\nUnder the Consumer Protection Act	The Consumer protection Act covers all goods and services including banking, e-commerce, telecom, insurance, electricity, transportation in the private and public sector.

Evaluation dataset of inheritance legal act with questions, context and response

Figure 72: Evaluation dataset for inheritance

Query	Context	Reference
What is the legal document that contains specific details about the distribution of an owner's estate after their death, according to the Indian Succession Act, 1925?	What is a will?\r\nUnder the Indian Succession Act, 1925, a will is a declaration or a legal document that contains specific details like the name of one or more persons who will acquire, manage, and get benefitted from an owner\u2019s estate after his/her death.	Will is the legal document that contains specific details about the distribution of an owner's estate after their death, according to the Indian Succession Act, 1925
What is a legal heir, according to the given definition?	Therefore a legal heir is an individual who takes the place of the property of his/her ancestor, either by law or by a will.\r\nImportance of heir\r\nIt is essential to identify a legal heir for every person owning property; they are the successors for property claims and insurance coverage.	a legal heir is an individual who takes the place of the property of his/her ancestor, either by law or by a will
Who is the legal heir of a female Hindu who dies intestate?	Mother of the mother\r\n* Father of the mother\r\nIX\r\n* Sister of the mother\r\n* Brother of the mother\r\nLegal heir of female Hindu\r\nThe property of a Hindu female dying intestate is to be transferred to:\r\n* The daughters and sons (this includes the children of any dead son or daughter)	The daughters and the sons, including the children of any dead son or daughter wiill be the legal heir of a female Hindu who dies intestate.
What was the change in the right share in ancestral property granted to daughters in India before the year 2005?	Rights of Daughters\r\nBefore the year 2005, the right share in the ancestral property was only given to unmarried daughters in India. However, in the year 2005, the equal rights and duties of a son were granted to daughters.	Before the year 2005, the right share in the ancestral property was only given to unmarried daughters in India. However, in the year 2005, the equal rights and duties of a son were granted to daughters.
What is the minimum information required to apply for a legal heir certificate?	Documents required for legal heir certificate\r\n* Deceased person\u2019s name\r\n* The relationship and the name of the applicant with the deceased person\r\n* Applicant\u2019s signature\r\n* Applicant\u2019s residential address	The minimum information required to apply for a legal heir certificate are: Deceased person's name, The relationship and the name of the applicant with the deceased person, Applicant's signature, and Applicant's residential address

[9.C] Results on the evaluation dataset

Results showing the LLM response and all the scores shown in the Chapter Results. Shown only for the case of simple retrieval

For Copyright:

Figure 73: Evaluation results for copyright

	Query	Context	Reference	LLM_response	Relevancy Score	Faithfulness Score	Semantic Similarity Score	Correctness Score
0	What is the primary purpose of copyright, acco...	It is a bundle of rights comprising of rights ...	The primary purpose of copyright is to ensure ...	To ensure certain minimum safeguards of the ri...	True	True	0.914001	4.0
1	What is the form that needs to be filled out t...	To apply for copyright, application for regist...	Form IV needs to be filled out to apply for co...	Form IV.	True	False	0.725553	4.0
2	What is the typical duration of copyright prot...	Copyright protection typically lasts for 60 ye...	Typical duration of copyright duration is 60 y...	60 years.	True	True	0.787101	3.0
3	What is the minimum punishment for copyright i...	Remedy for copyright infringement\r\nCopyright...	The minimum punishment for infringement of a c...	Imprisonment for six months with a minimum fin...	True	True	0.895927	4.0
4	What is the minimum rank of a police officer w...	Also, in case a copyright infringement has hap...	The minimum rank of a police officer who can s...	A sub-inspector.	True	False	0.704440	4.0

For divorce:

Figure 74: Evaluation results for divorce

	Query	Context	Reference	LLM_response	Relevancy Score	Faithfulness Score	Semantic Similarity Score	Correctness Score
0	What type of divorce is considered by the cour...	Divorce with Mutual Consent\r\nWhen the couple...	Divorce with Mutual consent	Divorce with Mutual Consent.	True	True	0.980539	4.0
1	What is the minimum duration of separation req...	Section 10A of Indian Divorce Act, 1869, requi...	The minimum duration of sepration required for...	Two years.	False	False	0.718859	3.0
2	What are the three aspects that couples need t...	maintenance and property rights need to be agr...	The three aspects that couples needto reach a ...	Alimony or maintenance issues, child custody, ...	True	False	0.791490	4.0
3	What types of expenses will be considered when...	Custody of child, alimony to wife and litigati...	Expenses considered while issuing a decree of ...	Custody of child, alimony to wife, and litigat...	False	False	0.924057	3.0
4	What is one circumstance that may favor the sp...	the failing health or a medical condition of o...	Failing health or a medical condition of the s...	The failing health or a medical condition of o...	True	True	0.904862	4.0

For consumer protection:

Figure 75: Evaluation results for consumer protection

	Query	Context	Reference	LLM_response	Relevancy Score	Faithfulness Score	Semantic Similarity Score	Correctness Score
0	What is the main law that provides protection ...	\ufeffConsumer Protection Laws in India\r\nCon...	Consumer Protection Amendement Act, 2002 is th...	The Consumer Protection Act.	True	True	0.838335	4.0
1	What is the primary objective of the Consumer ...	In this article, we look at the protection aff...	The primary objective of the Consumer Protecti...	To provide better protection of consumers and ...	True	True	0.922810	4.0
2	What is the procedure to file a complaint in c...	Procedure to File Consumer Case\r\nAny consume...	Any consumer complaint relating to a good or s...	Any consumer complaint must be filed in writin...	True	False	0.827317	4.0
3	What is the possible outcome if the complainan...	Hence, if during the proceedings, the complain...	If the complainant fails to appear on the date...	The District Forum may either dismiss the comp...	True	False	0.901145	4.0
4	What kind of goods and services does consumer ...	The Act covers all goods and services includin...	The Consumer protection Act covers all goods a...	All goods and services, including banking, e-c...	True	True	0.875117	4.0

For inheritance:

Figure 76: Evaluation results for inheritance

	Query	Context	Reference	LLM_response	Relevancy Score	Faithfulness Score	Semantic Similarity Score	Correctness Score
0	What is the legal document that contains speci...	What is a will?\r\nUnder the Indian Succession...	Will is the legal document that contains speci...	A declaration.	True	False	0.621223	3.0
1	What is a legal heir, according to the given d...	Therefore a legal heir is an individual who ta...	a legal heir is an individual who takes the pl...	An individual who takes the place of the prope...	True	True	0.912004	4.0
2	Who is the legal heir of a female Hindu who di...	Mother of the mother\r\n* Father of the mother...	The daughters and the sons, including the chil...	The daughters and sons (including the children...	True	True	0.814912	4.0
3	What was the change in the right share in ance...	Rights of Daughters\r\nBefore the year 2005, t...	Before the year 2005, the right share in the a...	Only unmarried daughters had the right share i...	True	True	0.795928	4.0
4	What is the minimum information required to ap...	Documents required for legal heir certificate\...	The minimum information required to apply for ...	The minimum information required to apply for ...	True	True	0.982581	4.0

Chapter 10 - Conclusion

None

Add checklist here