LexLLM: Personal AI Legal Assistant

A Project Report
Presented to

The Faculty of the Department of Applied Data Science
San Jose State University

In Partial Fulfillment

Of the Requirements for the Degree
Master of Science in Data Analytics

By

Abdul Sohail Ahmed,.; Saklecha, Sourab Rajendra; Sampath Kumar, Namratha; Khamker, Yuti Ashwin; Kutty Shivakumar, Swetha Neha

December 6, 2024

APPROVED FOR DEPARTMENT OF APPLIED DATA SCIENCE

_____

Dr. Simon Shim, Project Advisor

_____

Dr. Lee C. Chang, Project Advisor

ABSTRACT

LexLLM: Personal AI Legal Assistant

By

Abdul S. Ahmed, Namratha S. Kumar, Sourab R. Saklecha, Swetha N. K. Sivakumar, and Yuti A. Khamker

Purposes

With the overwhelming volume of legal codes, past cases, policies, and intricate legal frameworks, it has become increasingly difficult for everyday citizens to comprehend complex legislative codes and policy documents and stay updated on the constant amendments. It is also difficult for the lawyers if they want to access the information to decide on the approach they want to take for the particular case which is very similar to a previous one. Solutions enabling faster consumption and information retrieval of these documents have become vital to foster a more informed and engaged citizenry. The whole purpose of this project is to make people's lives easier by allowing faster access to the required legal information.

Tasks

This project aims to develop four intelligent Retrieval-Augmented Generation (RAG) based chat applications powered by large language models (LLMs) for efficient accessibility and processing of PDF documents that may contain any snippet of the federal legislative code, policies, and past cases of this country. Users would be able to load PDF files and seek answers by posing questions about the contents in a smooth chat interface. Project tasks involve identifying the documents from existing data sources containing legal information. These documents will need to be web-scraped and cleaned. The cleaned dataset was then modeled and stored in a vector database and used to fine-tune the LLMs. These LLMs will then be deployed

and made user accessible using a website. Evaluation will be carried out to compare the different LLMs. The entire project development cycle will follow the Agile approach. Improving the application's usability and efficacy in legal document interpretation was the main focus of this project.

Outcomes

The main outcome of the project would be improved performance of the existing ground LLMs which will be achieved using extensive fine-tuning and refinement. A comparison of the performance of different fine-tuned models will be performed using evaluation metrics like Perplexity, OpenAI Evals and Human Evaluations.

Applications

This RAG chat application will go beyond technical achievement and hold the potential to transform how legal documents are accessed and utilized by the general public. In the long run, it can help enhance the capability of normal citizens to make effective decisions by improving their legal literacy which helps to ensure proper compliance with different laws, regulations, and policies. This application can help in improving labor law and tax code compliance. It can also help to improve the policies followed by different companies to ensure more safety of the user data and ensure a better experience. Improvements gained in fine-tuning LLMs here could then be applied to other use cases.

ACKNOWLEDGMENTS

We are sincerely grateful to Dr. Simon Shim and Dr. Lee C. Chang for their constructive feedback, crucial guidance, and constant support throughout the project. We would also like to thank the Department of Applied Data Science for providing the necessary resources to perform different operations and ensure the successful completion of the project.

TABLE OF CONTENTS

Introduction

Project Background and Executive Summary

Project Background

People attempting to read and comprehend legal texts and systems face difficulties because they are always changing. Staying educated is crucial for individuals and businesses, whether regarding policy interpretation or legal modifications. By definition, legal documents are long, hierarchical, and replete with citations and specialist terminology (Turtle, 1995). People frequently feel overwhelmed by this intricacy. According to Thomson Reuters, attorneys spend about 60% of their time reading the more than 2.5 billion pages of papers produced by the legal industry each year. Even with the availability of legal summarizing tools, many of them cannot meet domain-specific requirements and offer useful assistance.

By creating a flexible legal chatbot using fine-tuned large language models (LLMs) and Retrieval Augmented Generation (RAG), this project seeks to close the gap. This chatbot will make legal information more approachable and useful for everyone, from regular people to legal professionals, thanks to its training on thousands of legal cases, research, textbooks, and legislation.

Needs and Importance of the Project

To ensure justice and well-informed decision-making, legal understanding is essential. Research shows that domain-specific training on legal datasets can greatly improve LLM performance by providing superior generalization across other datasets (e.g., Gallegos & George). It might be difficult for users to completely comprehend their rights, obligations, or potential hazards because current legal aids frequently fall short in simplifying the complex language and structure of legal materials.

This chatbot will provide succinct, precise answers to legal questions by utilizing RAG and optimizing LLMs on a large corpus of legal writings. It will make complicated legal systems easier to understand, which will improve decision-making, encourage compliance, and increase transparency. Additionally, by meeting a variety of user demands, such as comprehending laws, court decisions, or regulatory requirements, the project fills a significant vacuum in legal support.

Target Problem

The goal of this chatbot is to make legal advice more approachable and useful for everyone by streamlining the difficulties associated with comprehending and navigating complicated legal material. To guarantee precise and pertinent answers, it makes use of state-of-the-art large language models (LLMs) that have been refined over thousands of court cases, policies, textbooks, and laws, aided by RAG. The chatbot assists in distilling complex legal documents into concise and understandable summaries, including corporate policy, court decisions, and regulatory regulations. In addition to giving consumers a grasp of previous court cases to aid in comparison and decision-making, it offers insightful information about all laws, rights, regulations, and other legal frameworks.

The chatbot enables people and businesses to make educated decisions and guarantee compliance by deciphering terms of service and privacy rules. This application fills the knowledge gap between legal language and common sense, whether you're a business owner, a legal practitioner, or someone who wants to understand their rights.

Motivation and Goal of the Project

Accessible legal comprehension is desperately needed as a result of the convergence of technology and law. People frequently struggle to comprehend their rights and responsibilities due to a combination of economic limitations, language hurdles, and legal complications. To lower risks and preserve trust, organizations also need to manage regulatory environments.

This project compares and refines models such as OpenAI (GPT-4o), Llama 3.1, Mixtral 8x7b, and Gemini 1.5 in order to give a dependable and easily accessible solution. With an emphasis on accuracy and clarity, these models' capacity to summarize legal texts will be assessed. The knowledge acquired will be helpful to researchers and legal professionals as well as regular people looking for clarity and direction in legal issues.

Project Approaches and Methods

Several methodical procedures were followed during the chatbot's creation to guarantee its dependability and efficacy. The first step in the process was data collection, which involved gathering legal documents from reliable sources, such as laws, court decisions, and company policies, and understanding legal jargon through some textbooks. After that, the data was preprocessed to ensure correctness and consistency throughout the dataset by standardizing the text. Following processing, the data was pushed to Astra DB, a powerful vector database solution that makes it possible to store and retrieve legal documents effectively. This allows for prompt and precise answers to RAG-based requests. To improve their capacity to process domain-specific content, pre-trained LLMs like OpenAI GPT 4o, Llama 3.1, Mixtral 8x7b, and Gemini 1.5 were refined using legal datasets for model training. To guarantee dependability and usability, the chatbot's performance was carefully assessed using metrics including accuracy,

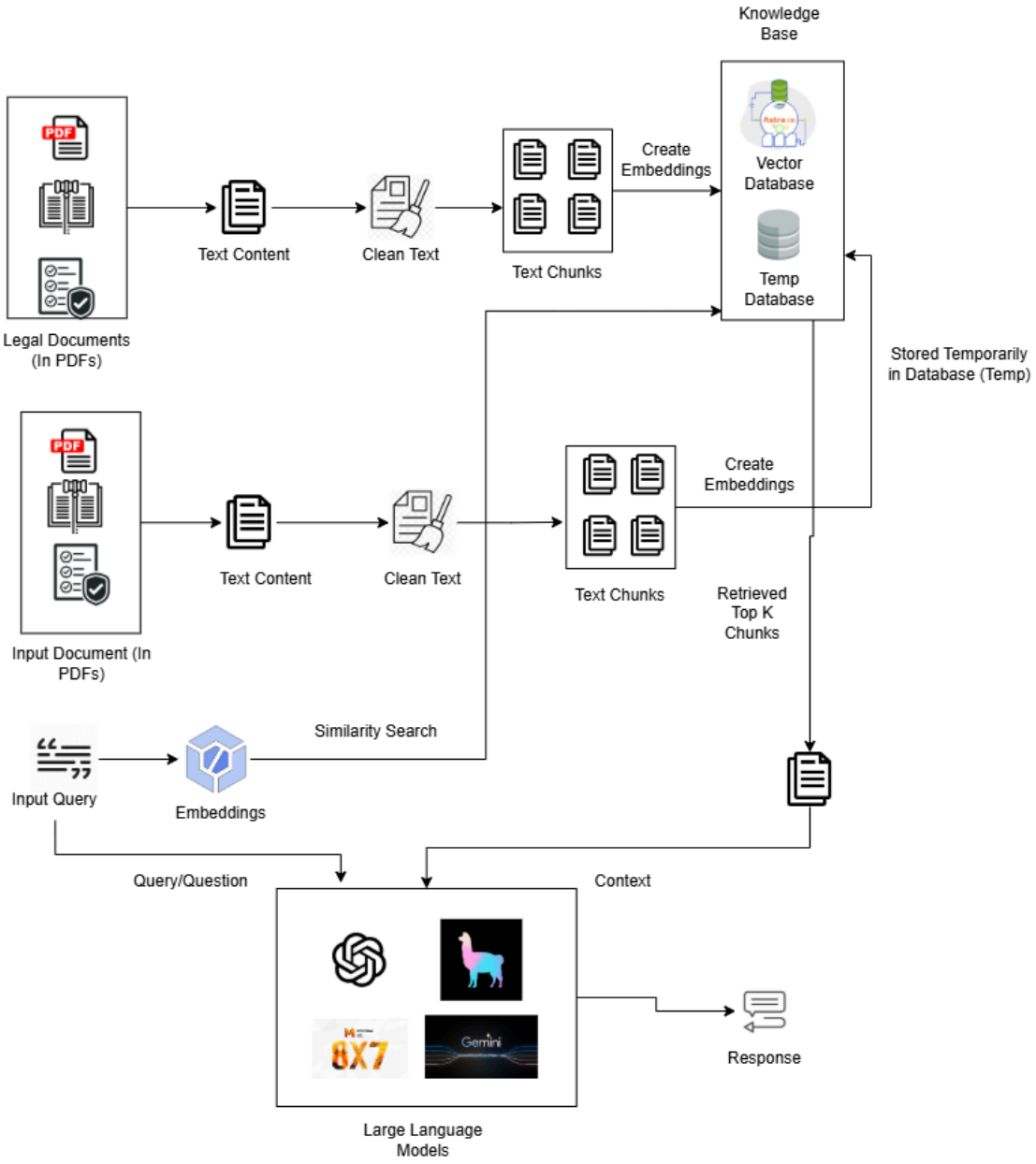
perplexity, and human assessments. Following validation, the models were implemented in an intuitive chat interface that facilitates easy engagement with lawful content. The chatbot will get routine maintenance, such as updates and assistance, to ensure that it continues to offer accurate and current legal advice and to remain relevant and effective.

Expected project contributions and applications

This study intends to analyze the usefulness of employing RAG and fine-tuning LLMs to enhance the performance of legal chatbots. The study will offer insights into how these methods enhance the precision, applicability, and understanding of legal texts by including domain-specific datasets. Because the results demonstrate how AI can simplify complex legal knowledge, they will be useful to researchers, legislators, and legal practitioners. It will also highlight the best ways to use RAG and fine-tuning to maximize AI technologies in specific fields. Figure 1 shows the proposed work of this project.

Figure 1

Workflow Diagram with RAG implementation



Note. Illustrating the steps involved in the workflow process with responses to the queries are generated using RAG-based and Fine-tuned LLMs. Original from Team 1.

Project Requirements

Functional Requirements

This project develops an intelligent legal chat application that provides relevant and accurate answers by incorporating a Large Language Model (LLM) and Retrieval Augmented Generation (RAG) architectural approach. The user will log securely into the application using Google OAuth. As the core of the application, any legal pdf documents can be submitted along with the questions through the application. The major functional requirements were to make sure the LexLLM application responded to the question accurately by identifying keywords and ranking the suggested answers based on relevance and quality of answers. The application is integrated with a vector database to store the vector embeddings of the legal documents and based on similarity metrics the answers will be retrieved. The result of this was to ensure the required infrastructure to store and retrieve the documents efficiently. Further, this project utilizes LLM to generate answers from the knowledge base. The implementation of the LLM requires high computational power in terms of Graphics Processing Unit (GPU) and Tensor Processing Unit (TPU). For the efficient operation of the LLM, the parameters learned during the training phase need to be stored in the memory for quick access during the inference phase. For the preprocessing of the data, an Integrated Development Environment (IDE) will be used to execute the data transformation.

AI Requirements

The LexLLM application has Natural Language Understanding (NLU) capabilities to interpret the questions submitted by the user. The implementation involves the best-suited LLM such as Llama 3.1, Gemini 1.5, OpenAI (GPT-4o), and Mixtral 8x7b for generating the answers to the legal questions. The Llama 3.1 model is an open-source LLM from Meta and the models

are available in different parameters like 7 billion, 13 billion, and 70 billion. It was trained on 2 trillion tokens and employs reinforcement learning from human feedback to ensure helpfulness. The GPT model from OpenAI is based on Transformer-Decoder architecture and is renowned for its adaptability and capacity to produce human-like responses. Gemini is a Transformer and Mixture-of-Experts (MoE) architecture model and is highly efficient in training. Based on Transformer architecture, designed for multi-turn dialogue and trained to support legal resources, it is great for answering follow-up questions and maintaining context across multiple questions. Llama is an open source LLM model developed by Meta and is based on Longformer architecture which extends the Transformer architecture. Fine-tuning models with legal data fine-tuning can address a broad range of legal queries and provide the user with insightful responses. With fewer parameters and computational resources, Mixtral effectively processes long data and achieves impressive results. With the help of legal data, the models were trained to comprehend the text's structure and semantics and to produce responses that were both logical and pertinent to the context.

Data Requirements

The Legal question-answering application requires the legal articles for training and fine-tuning the LLM models. One of the main sources of data was legal proceedings from USCode and Westlaw, two governmental websites. The documents provide the models with a broad range of legal material to learn from, including case law, statutes, regulations issued, and legal opinions. The documents related to legal cases were downloaded manually to ensure that only the relevant ones are used, and policies adopted by different companies and terms of use and service provided by companies were extracted from the GitHub repository where they have scrapped the data from various websites. Information on court cases brought for breaking laws

pertaining to technology, regulations adhered to by various tech companies, and terms of service of websites and apps offered by the tech companies to enhance the capacity to handle legal issues pertaining to the tech industry. The data was extracted in the PDF format consisting of the rules, acts, and resolutions, and the Code of Laws of the United States. The size of the data was substantial, given the extensive nature of legal documents available on these platforms. To give more contextual information, the data goes through an extensive pre-processing stage that includes cleaning, formatting, and additional metadata.

Project Deliverables

The LexLLM project is implemented in the CRISP-DM methodology consisting of a broad spectrum of outcomes to achieve in this project. Detailed knowledge of the business requirements and objectives (Business Understanding), data sources (Data Understanding), carefully prepared data ready for further model building (Data Preparation), well-built models customized to address the business problem (Modeling), accurate metrics to assess model performance (Evaluation), and a smooth deployment plan (Deployment) are the deliverables of the project. Each deliverable is crucial for the success of the project.

Project Proposal

The Project proposal outlines the overall background of the project, the challenges, and the opportunities that necessitate the chosen project. It has a brief description of the proposed problem, the goals, and the initial framework to implement the solution. In this, the existing research was evaluated to identify the scope of improvement.

Project Planning

Project planning was carried out using the Jira application, which employs a specific project management hierarchy consisting of Tasks, User Stories, and Epics. Here, the work was

completed through different Sprints and there was a set number of days for each sprint. Six Sprints, each lasting two weeks, make up the current project hierarchy. Various Jira plugins were used to build the Gantt chart, and Work Breakdown Structure that helped in performing effective Project Scheduling.

Work Breakdown Structure. It is a tool for project management that arranges the list of activities that have to be completed as per the project's hierarchy. The project's Work Breakdown Structure, which is composed of the primary tasks of the six Epics and is further broken into related User Stories and Sub Tasks, was created using the Jira plugin Move and Organize.

Gantt Chart. The list of tasks to be completed for the project is represented as a bar chart, with information on each task's start date, end date, duration, allocated person, and dependencies between the tasks. The project creates a Gantt chart that shows the timelines of the tasks and the resources involved in them using WBS Gantt - Chart, a Jira plugin.

Data Collection

The legal documents from governmental websites such as USCode and Westlaw were extracted. Law opinions, legislation, regulations, and case laws were included in the extracted data, which is in PDF format. The legal cases were downloaded manually. The policies, terms of use, terms of service, etc were extracted from the GitHub repository where they have scrapped the data from various websites. The terms of use and services include different terms mentioned by the companies that need to be followed when using applications or websites, policies include the details about the specific policy related to that company, and legal cases contain comprehensive information about the case that includes opinions made by each party, factual background, and the allegations made related to class actions.

Prototype

A prototype solution was developed using RAG architecture for analyzing the responses of the legal application. The prototype was built based on the LLM Open AI and OpenAI embedding. The prototype aimed to demonstrate the feasibility and effectiveness of question-answering in the legal domain and served as the foundation for further development.

Data Engineering

The text data from the legal document is the extracted raw data. To get rid of any outliers or superfluous data, it is cleaned and processed. The Python data frames were used to sanitize the data. Several natural language processing techniques, including tokenization, stemming, and lemmatization, were used to process legal documents. The final cleaned data is put into the Large Language models by dividing it into Train, Test, and Validation sets.

Model Development and Testing

The model development process for this project uses a combination of cutting-edge large language models (LLMs) to enhance the quality of answers for legal questions using RAG architecture. The LLMs employed are Llama 3.1, OpenAI (GPT-4o), Gemini 1.5, and Mixtral 8x7b because of their renowned natural language understanding capabilities, robustness, and performance in legal text processing. Further, all the models were fine-tuned for precise replies in the context of law.

Performance Evaluation

The performance evaluation involves assessing the effectiveness of the response to the legal questions by different LLMs developed. The metrics Perplexity, Open AI Evals, and human evaluations were used in the assessment process.

Optimal Model Selection

Optimal model selection is crucial for the development of any applications involving artificial intelligence including the LexLLM. The best optimal model was evaluated based on performance, model complexity, computational resources, and scalability.

Production Application

LexLLM application is built using the open-source Python framework designed to build interactive applications. The backend of the application was integrated with the AstraDB vector database to effectively retrieve the relevant legal documents. The optimal LLM model then generates precise responses to the user's requirements based on the relevant documents extracted from the vector database.

Project Report

The project report provides an overview of the project goals, the research on the problem statement, the complete implementation, the evaluation of the LLM, and the production application. Table 1 outlines the project deliverables and their associated due dates.

Table 1

Project Deliverables and the Due Dates

Deliverable	**Due date**
Project Proposal	02/15/2024
Data Collection	02/29/2024
WBS & Gantt chart	03/13/2024
Data Engineering	03/14/2024
Model Development and Testing	04/18/2024
Performance Evaluation	09/18/2024

Deliverable	Due date
Optimal Model Selection	10/07/2024
Production Application	12/06/2024
Final Report	12/06/2024

Note. The above table lists the deliverables that are/will be provided throughout the project. Original from Team 1.

Technology and Solution Survey

Open AI Team (2024) reported the development of the state-of-the-art model GPT-4. It is capable of processing multiple kinds of input data like text, images, etc which is why it is called a multimodal model. It demonstrated a performance very similar to that of the human, such as achieving the score in a simulated scenario of a bar exam which placed it in the top 10%. Its architecture is very similar to that of the previous models like GPT-3.5. Still, here it incorporates the concept of reinforcement learning from human feedback (RLHF) which helps it to improve its performance. It outperforms all the previous models by attaining high scores in benchmarks like MMLU (86.4%). The model has strong multilingual capabilities which helps it surpass the English-language SOTA system in 24 languages among 26 that are tested which includes the ones with limited resources like Welsh. Despite a lot of advantages, GPT-4 still has some limitations that are carried from the previous models like hallucination, biases, errors when solving reasoning problems, etc. Some processes like Safety mitigations are incorporated to reduce the scenarios of incorrect responses concerning sensitive content by close to 82% as compared to the previous model GPT-3.5.

Brown et al. (2020) explore recent developments in natural language processing (NLP) by evaluating the effectiveness of pre-trained language representations in a variety of NLP tasks. An array of autoregressive language models known as GPT-3 is thoroughly trained using a

meticulous selection of high-quality datasets, including filtered versions of Common Crawl and a curated corpus. The evaluation included GPT-3 models of different sizes, ranging from GPT-3 Small (125 million parameters) to GPT-3 175B (175 billion parameters). The evaluation of the model is done against a wide range of NLP benchmarks, with a particular focus on methodologies such as few-shot, one-shot, and zero-shot learning. Among the various model sizes, GPT-3 175B performed the best. In evaluation with regards to word manipulation, it achieves a significant few-shot accuracy, including 66.9% on random insertions as well as high scores on anagrams and letter cycling. It achieves a 65.2% accuracy in SAT analogies rate. In the zero-shot setting, GPT-3 175B achieves about 52% accuracy in news article generation, showcasing the ability to generate content that resembles human-written articles. These results demonstrate GPT-3's effective comprehension of language and generation capabilities across a wide range of tasks and scenarios. RAG (Fine-tuned, Open-Domain) outperforms existing SOTA results on three Open-Domain QA tasks: NaturalQS (44.5), WebQS (45.5), and TriviaQA (68.0). GPT-3 still has problems with text synthesis like common sense reasoning, and scaling up models may cause challenges in pre-training objectives, sample efficiency, and bias retention.

Radford et al. (2019) evaluate the performance of language models trained on web-based text data, particularly GPT-2 models of various sizes (117M, 345M, 762M, and 1542M parameters), across several natural language processing tasks in a zero-shot setting. With benchmark datasets like CoQA, GLUE, decaNLP, WMT-14 Fr-En, CNN, Daily Mail, and Natural Questions, tasks are evaluated ranging from conversational question answering to machine translation. With a large language model encompassing 1.5 billion parameters, competitive results are obtained when compared to baseline methods, including an F1 score of 55 on CoQA and better GLUE metrics. Language modeling perplexity scores ranged between

35.13 and 8.63, showing strong competence. Summarization tasks yielded ROUGE F1 scores of up to 29.34, while translation tasks generated BLEU values ranging from 5 to 11.5. Question-answering tasks showed high accuracy and F1 scores, with an exact match score of 4.1% and an accuracy of 63.1% on confident answers. These findings emphasize GPT-2 models as versatile multitask learners, capable of excelling across a wide range of NLP tasks. GPT-2's ability to generalize efficiently to many different tasks without the need for task-specific fine-tuning makes it a desirable choice for applications that need comprehensive language understanding and generation capabilities.

To address the need for a SOTA that can handle multimodal inputs, Rohan et al.(2023) and their team at Google introduced Gemini- Ultra, Pro, and Nano series. Gemini has now achieved human-expert performance on the well-studied exam benchmark MMLU and is setting new SOTA benchmarks in every one of the 20 multimodal benchmarks. To train Gemini, the team used a dataset that is both multimodal and multilingual. The pretraining dataset used data from web documents, books, and code, and included image, audio, and video data. The Gemini models are trained on TPUv5e and TPUv4. A sequence length of 32,768 tokens was used to maximize their context length efficiency. They found that the ultra model retrieves the correct value with 98% accuracy in cases of queries with full-length context. For text, Gemini Ultra outperformed all existing models, achieving an accuracy of 90.04% when used in combination with a chain-of-thought prompting approach. Gemini Ultra can solve 32% of the questions compared to the 30% that GPT-4 got on the same test. On the MMLU benchmark, which is the de facto measure of progress for LLMs, Gemini Ultra scores more than 90%. The team wants to work on the hallucinations on the model and further reduce the need for fine-tuning and grounding.

Gemma Team, Google DeepMind (2024) highlights the need for open-source lightweight LLMs to perform similar tasks like chatbots and summarization. They introduced Gemma to address this need. Gemma is a light-on-resource consumption version of Gemini and was trained on 6 trillion tokens of text, primarily in English, sourced from web documents in mathematics and code. Similar to Gemini, this model combines cutting-edge comprehension and reasoning abilities at scale with high generalist ability in text domains. The transformer decoder (Vaswani et al.) serves as the foundation for the Gemma model architecture. An 8192 token context length is used to train the models to employ rotary positional embeddings in each layer instead of absolute positional embeddings, and use the GeGLU activation function instead of the ReLU activation function. Gemma was assessed using both automated benchmarks and human evaluation in a wide range of categories. In creative writing activities, coding, and following instructions, Gemma 7B IT has a 51.7% positive win rate and Gemma 2B IT has a 41.6% win rate over Mistral v0.2 7B Instruct. These results are based on a held-out sample of roughly 1000 prompts. Gemma performs better on 11 out of 18 text-based tasks than comparable sized open models. Findings on MMLU (64.3%) and MBPP (44.4%) show that Gemma continues to execute at a high level and that there is still headroom in publicly accessible LLM performance. To develop reliable models that function as planned, more research is required. Factual accuracy, alignment, sophisticated reasoning, and resilience to hostile input are a few examples of further research fields.

Jiang et al. (2023) introduced a new model named Mistral 7B. It is a language model that contains 7 billion parameters and is specifically designed to showcase high efficiency and performance. It easily surpassed its competitors LLaMA 2 and LLaMA 1 in the case of performing reasoning, solving mathematical problems, and efficient code generation. There are

some architectural differences like the inclusion of a grouped-query attention mechanism to ensure faster inference with the presence of a sliding window to handle the data containing long sequences with less amount of computational cost. It attained high performance in some of the NLP benchmarks like common sense reasoning where it surpassed LLaMA 2 on the datasets like Hellaswag. It achieved higher accuracy in solving math problems and performing coding-related tasks as compared to Code-LLaMA 7B. This is demonstrated in numbers where it scored 52.2% on the GSM8k dataset and 13.1% on MATH with their corresponding settings of 8 and 4-shot learning. Its fine-tuned model named Mistral 7B-Instruct is able to surpass all the models based on 7 billion parameters and it is very close to performance with their 13 billion parameters counterparts. Even though the size of the model is small it shows performance and efficiency very similar to models that are thrice its size. It incorporates the safety mechanism to ensure content moderation and the generation of safe responses.

Jiang et al. (2024) launched Mixtral 8x7b. It is a language model containing Sparse Mixture of Experts (SMoE) layers. It achieved state-of-art performance while ensuring that the computational efficiency is maintained. This model uses only 13 billion of its 47 billion parameters to process each token of the text content. It is pre-trained on various types of multilingual datasets which is present in The Pile paper (Gao et al., 2020) that contains academic publications, technical papers, and easy-to-use general-purpose datasets that are mainly focused on multilingual content. This helps to enhance the capabilities of the model to handle tasks present in languages like French, German, Italian, and Spanish, along with English. It surpassed models like LLaMA 2 70B and GPT-3.5 across various benchmarks. Some of them are common sense reasoning-related tasks where it attained 84.4% on the Hellswag dataset and 83.6% on PIQA; mathematics tasks where it achieved 74.4% on the GSM8k dataset and 28.4% on the

MATH dataset; and general knowledge tasks like MMLU where it attained the score of 70.6%. The fine-tuned Mixtral 8x7b-Instruct surpassed Claude 2.1, GPT 3.5 Turbo, and LLaMA 2 70B in the benchmark of human evaluations on the tasks related to instruction-following and effective code generation.

By training on more tokens than normal, Touvron et al. (2023a) aim to attain optimal performance at different inference budgets by utilizing a range of language models, including LLaMA, with parameters ranging from 7B to 65 B. Recent studies have shown that smaller models trained on more data have outperformed within a limited computational cost. LLaMA consists of the foundational models ranging from 7B to 65B parameters and the model is trained on trillions of tokens using English CommonCrawl, C4 dataset, GitHub, Wikipedia, Gutenberg and Books3, arXiv, and Stack Exchange. Thus using various datasets leads to covering diverse domains in the research. The architecture of the model is based on transformer architecture with modifications in pre-normalization, SwiGLU activation function, and rotary positional embeddings. Further, the models are fine-tuned hyperparameters to achieve the goal of the experiment. Llama outperformed GPT-3 on most benchmarks, especially in code generation, and LLaMA 13 B outperformed LAMBADA 137B on both HumanEval and MBPP by developing models that can run at high-performance levels on a single GPU, the research is aimed at increasing access to LLMs.

The study “LLAMA 2: Open Foundation and Fine-tuned Chat Models" by Touvron et al.(2023b) aims to enhance LLaMA 2 for various natural language understanding tasks focusing on computational efficiency during inference and exploring the distinction between open-source and closed-source models. In this research, a huge volume of personal data about private individuals summing to 2 trillion tokens of data is used. Bytepair encoding(BPE) algorithm is

employed as a tokenizer with a total vocabulary of 32k tokens. The pre-training is performed on Meta's Research SuperCluster (RSC) using reinforcement learning algorithms such as Rejection Sampling and Proximal Policy Optimization (PPO) for fine-tuning. The LLaMA2 model shows no sign of saturation even following pre-training on 2 trillion tokens. LLaMA2 outperforms the MosaicML NLP Team model and Mixtral model on most benchmarks. In the examination of instructional tuning, the study emphasizes how crucial high-quality data is to match conversational-style instructions to LLM alignment. LLaMA 2 has shown significant performance improvement over both open-source models as well as proprietary models on evaluation. Table 2 provides a comparison of different existing technologies.

Table 2

Comparison of Technology Survey

Authors	Dataset	Models	Results	Conclusion
OpenAI Team (2024)	Common Crawl, Refined Web	GPT-4	MMLU (86.4%), Hellswag (95.3%) GSM8k (92%), and the top 10% in the bar exam	GPT-4 provides multimodal capability and better comprehension of data.
Brown et al. (2020)	Common Crawl, curated corpus	GPT-3 Small to GPT-3 175B	66.9% accuracy in word manipulation, and 65.2% in SAT analogies. RAG outperformed SOTA in Open-Domain QA.	GPT-3 performs well in terms of language comprehension and generation.
Radford et al. (2019)	CoQA, GLUE, decaNLP, WMT-14 Fr-En, CNN	GPT-2 (117M, 345M, 762M, 1542M)	F1 score of 55 in CoQA. Perplexity ratings ranged between 35.13 and 8.63.	Efficient in performing multitask learning without task-specific fine-tuning

Authors	Dataset	Models	Results	Conclusion
Anil et al.(2023)	Native multimodal web data	Gemini	Accuracy (90.4%)	Gemini Ultra surpasses human-expert performance on the exam benchmark MMLU, scoring >90.0%
Gemma Team, Google DeepMind (2024)	Native text web data	Gemma	MMLU (64.3%) and MBPP (44.4%)	Gemma outperforms similarly sized open models on 11 out of 18 text-based tasks.
Jiang et al. (2023)	Crawled from Open Web	Mistral 7B, Mistal 7B-Instruct	MMLU(60.1%), Hellswag(81.3%), PIQA(83%) GSM8k(52.2%), and MATH (13.1%)	Excel in tasks requiring context awareness and long-term memory like code generation, and QA
Jiang et al. (2024)	Pile Paper dataset	Mixtral 8x7B, Mixtral 8x7B-Instruct	Hellswag (84.4%), PIQA (83.6%), GSM8k(74.4%), and MATH (28.4%)	Efficiently handle tasks in different languages. Fine-tuned model better in instruction-following tasks
Touvron et al. (2023a)	English commonCrawl, C4 dataset, Github	LLaMA	LLaMA-65B shows 68.2% accuracy in zero-shot exact matches.	LLaMA outperforms GPT-3 being 10x smaller. LLaMA has the lowest carbon emitted compared to OPT and BLOOM
Touvron et al. (2023b)	Personal Information	LLaMA2	LLaMA2 achieved 60% on prompts, and LLaMA-2-Chat 34B achieved 75% on the overall win rate.	LLaMA 2 competes closely with GPT3.5 and outperforms MPT and Mixtral models.

Note. Summary of Technology and Solution Survey. Original from Team 1.

Literature Survey of Existing Research

Adams et al. (2024) emphasize the need for an in-depth study of large language models (LLMs) in healthcare by focusing on their ability to interpret long clinical records. The dataset is

a collection of 20 hypothetical patient cases each of them with discharge summaries ranging from 5,090 to 6,754 words curated by expert physicians explicitly for benchmark. Nine open-source LLMs like Mixtral-8x7B-Instruct-v0.1 and GPT-3.5 Turbo, are tested for their capacity to perform the benchmark tasks. The evaluation focused on the models' performance through the required tasks, with the focus being on the metric accuracy and their ability to extract relevant information from the extensive clinical records. Mixtral-8x7B-Instruct-v0.1 stood out among the evaluated models, exhibiting an impressive accuracy rate of 73% in the task of information retrieval from single patient documents and 71% in the task of extracting accurate information from a mixture of documents belonging to different patients. It also exhibited the highest accuracy in comparison to others for correctly identifying when information is not readily available but the highest accuracy is only about 29% making it a significant challenge. Further research is needed to explore alternate prompting strategies, including models with shorter context lengths. The study's dependence on open-source models limits the scope of evaluation, and the use of multiple-choice questions may not accurately reflect real-world clinical circumstances.

Jain et al. (2020) discuss the challenges associated with creating large-scale datasets for document-level information extraction (IE). SCIREX, an efficient document-level IE dataset containing tasks like entity recognition and N-ary relationship extraction from scientific articles, was introduced. It is fully annotated with entities, mentions, coreferences, and document-level relations, making it an efficient resource for research in IE. A neural model is built that performs IE tasks on a document level. Preprocessing steps include PDF preprocessing, tokenization using SpaCy, and automatic labeling utilizing a BERT+CRF model trained on the SCIERC dataset. SciBERT for token encoding and a BiLSTM for document-level understanding. Mention

identification and categorization are carried out using a CRF tagger trained on BERT-BiLSTM embeddings. The model is then compared with DYGIE++ and DocTAET, the baseline models. The metrics such as Precision, Recall, and F1 are used for Mention Identification, Salient Mentions, and 4-ary Relation Extraction. The model achieves F1 scores of 0.712 for Mention Identification, 0.579 for Salient Mentions, and 0.611 for 4-ary Relation Extraction outperforming the baseline models. These results highlight SCIREX's importance in improving document-level IE. The developed model also serves as a baseline. While the dataset offers valuable insights, its domain-specific nature and biases concerning annotation could limit generalizability.

Medeiros et al. (2023) discuss enhancing accessibility to information found in standard vehicle manuals using Large Language Models (LLMs) and AI-powered chatbots. The dataset used consists of PDF extracts from a Ford Fiesta 2015 manual which is an automobile manual. Textual content is extracted from PDF documents and split into more manageable chunks. Three LLM-based chatbot approaches are evaluated Doc ChatBot uses LlamaIndex, LangChain libraries, and OpenAI API, Ask Your PDF uses LangChain library and OpenAI API, and Question and Answer System uses LangChain library, OpenAI API, and Sentence Transformers library with KPIs The metrics used to assess was based on response accuracy, cost efficiency, and user experience. Doc ChatBot has medium response accuracy, a high cost, and a bad user experience. Ask Your PDF has a high response accuracy, a medium cost, and provides a good user experience. The Question and Answer System has low response accuracy, low cost, and a good user experience. Various prompts, like zero-shot, one-shot, and few-shot, are used to evaluate the chatbot's effectiveness when accessing vehicle manuals across different questions. In all, "Ask your PDF" performed the best across metrics, establishing a balance between accuracy, cost, and user experience, while also performing well at zero-shot prompts and

providing an intuitive interface. There exist challenges in the interpretation of visual aspects, limited compatibility with the format of the PDF, and a consistent need for accuracy improvements with complex documents. Future attempts should focus on improving the chatbot's ability to effectively handle complex documents while improving user experience.

Saad-Falcon et al. (2023) propose the PDFTriage strategy to address the issues of document question answering (QA), to improve the ability of large language models (LLMs) to extract information from structured documents such as PDFs, web pages, and presentations. The dataset is from a sample of documents in Common Crawl consisting of 82 documents with 908 questions across various categories. PDFs are converted into structured metadata representations and are grouped into questions based on type. PDFTriage is compared to baseline approaches i.e., page retrieval and chunk retrieval, measuring its performance across different question types and document formats to assess its ability to provide correct and comprehensive answers. The evaluation metrics used are readability, informativeness, clarity, and accuracy. Overall Quality is used to assess the best-performing model, PDFTriage surpassed the baseline models in terms of overall quality and accuracy for various question types with 3.9 and 3.8 respectively. It exhibits a superior performance associated with document question answering, and uniformity across document lengths and formats. However, its application may be limited due to complex document architecture and data quality. Its performance may be affected by the quality of document metadata and underlying language models, needing continual refinement.

The research by Miller (2019) introduced a new machine learning architecture for lecture summarization service using the BERT model for generating text embedding, and K-means clustering for recognizing the sentence most representative of the overall context of the lectures. The study addresses the limitation of the traditional approach in text summarization and proposes

an advanced approach using transformer-based models. The lecture transcript from Udacity focusing on Health Informatics and reinforcement learning lectures are utilized as the source data. A combination of multiple tokenization techniques is implemented before the data is provided to the model. This research architecture uses BERT text embeddings to generate word embeddings that accurately determine the sentence in the lecture transcript. Further K-means clustering determines the sentence closest to the centroid of the lecture. A command line interface gives the user control for the user to upload lectures and the number of sentences for summarization. The limitations of this research are handling context in long lecture transcripts and automatically determining the number of sentences to be used for generating a summary.

"MufassirQAS," an LLM developed by Alan et al. (2024) attempts to deliver reliable responses to questions about religion while predominantly avoiding misleading information and preserving religious values. The demonstrated approach increases the accuracy and transparency in LLM using the Retrieval Augmented Generation (RAG) approach. The core of the project addresses the learning and understanding of religions due to the complexity of the written body of the religious teaching. The Proof of Concept (POC) is implemented in Google Collab using Chroma as a vector database and Langchain as an RAG toolkit. The final implementation is done on an open source platform “Flowise " that already possesses RAG capabilities built on Langchain hosted on Hugging Face cloud-host provider. The results regenerated by MufassirQAS outperforms the answers generated by ChatGPT. This research demonstrates that by utilizing vector databases and passing accurate word embedding techniques MufassirQAS achieves the primary goal of generating relevant answers at a time also respecting the principles of religion. The future scope of this research is to build smaller and more sustainable models that are efficient in generating answers and enhancing scalability.

Chen et al.(2023) developed EQUALS, a Legal Question Answering (LQA) dataset consisting of 6914 questions, articles, and answer triplets. This dataset covers 10 collections of Chinese Laws and a repository containing 3081 law articles with an average of 300 articles per law. This research proposed an approach to answer real-world questions by the retrieval of relevant Law articles and Machine Reading Comprehension Tasks. The retrieval of relevant articles was achieved using BM25 and a deep learning model Sentence-BERT. For Machine Reading Comprehension(MCR) Bidirectional Encoder Representations from Transformers (BERT) models are considered. This research primarily focused on providing a solution to the drawbacks of not having a large-scale high-quality LQA dataset. The diverse and complex nature of the law articles enhances the difficulty of comprehending tasks due to the intricate legal questions from the users. The future scope of the project suggests extensive research in LQA based on the genuine and well-annotated EQUALS dataset.

Topsakal et al.(2023) discuss utilizing Large Language Models (LLM) for rapid application development with LangChain libraries. LLMs have gained attraction withs tasks like essay composition, code writing, explanation, and debugging to expedite the creation of custom AI applications. LangChain has gained tremendous recognition in the AI community for its ability to interact with various applications. This research points out the significant achievements in AI over the past decade especially in Natural Language Processing(NLP). The development of generative AI models like BERT, GPT, and T5 revolutionized text generation, text summarization, question answering, sentimental analysis, and more. These achievements emerged as a powerful tool for the machine to generate human-like responses based on the training of large text corpora. GPT-1, GPT-2, and GPT-3 have showcased remarkable performance improvements in increasing the tokens despite the limitations on hallucination.

While LLM offers extensive capabilities in various tasks the future scope is to accurately generate responses without erroneous results. This limitation is further countered by the emerging adoption of LLM through LangChain among zillions of users addressing some of the limitations for application development.

In their paper about using LLMs as tax attorneys, Nay et al.(2024) talk about the requirement of intense logical reasoning and math skills to automate the validation pipelines in a manner relevant to the real-world economic lives of citizens and companies. To accomplish this, they utilized the "bypass_retrieval" experimental retrieval setup, initially establishing a baseline for evaluating the effect of retrieval and LLM knowledge. For the second retrieval experimental setup, 'Similarity_search,' was utilized to introduce potentially pertinent legal content into the prompt. Every OpenAI model was applied. The most capable GPT-3 model "Davinci" was used. A previous iteration of the GPT-3.5 called "text-davinci-002" is trained with supervised fine-tuning instead of reinforcement learning. The most capable GPT-3.5 model called "get-3.5-turbo" was also used. They evaluated by answer comparison from GPT 4. Accuracy was tested for all LLMs on four grounds: bypass_retrieval, lecture_notes, similarity_search, and gold_truth. The results show that LLMs can perform at high accuracy levels but not yet at expert tax lawyer levels, especially when paired with prompting upgrades and the appropriate legal texts. As LLMs develop, the legal profession and AI governance may be significantly impacted by their capacity for independent legal reasoning. Future research might assess the differences in performance between language models that have been specially trained and optimized for legal reasoning and LLMs that have been pre-trained.

According to Louis et al. (2023), current techniques for legal question answering (LQA) have a limited scope because they are either restricted to certain legal fields or only allow for

succinct, uninformative responses. To address this, they provide a comprehensive approach that uses a "retrieve-then-read" pipeline to produce long-form responses to any statute law queries. They created and published the Long-form Legal Question Answering (LLeQA) dataset, which consists of 1,868 expert-annotated legal questions in French, to complement this strategy. The first step in the data-building process was gathering excellent question-answer pairs related to a legal topic. After gathering roughly 2,550 legal questions, they added 4,183 articles from 34 legal acts—laws, decrees, and ordinances that are commonly referenced as supporting references but are not included in the original collection. A tiny subset of legislative articles—some of which are pertinent to the question—are chosen using a retriever. Next, a generator bases its response on the subset of articles that the retriever has returned. They evaluate retrieval performance, generation quality, and rationale accuracy as their three main evaluation criteria. As metrics, they employ F1, accuracy, and METEOR scores. They utilized a dense retriever that was optimized using only 1.5k in-domain samples, and the results showed that domain adaptation is crucial for improving performance as it surpassed robust retrieval baselines. In the future, the team wants to better deal with the hallucinations encountered by their model.

Bhattacharya et al. (2019) experimented with a large set of Indian Supreme Court judgments and a large variety of summarization algorithms, including both supervised and unsupervised ones. They claim that to their knowledge, there hasn't been any systematic comparison of the performances of different algorithms in summarizing legal case documents. From the Westlaw India website, they gathered 17,347 Supreme Court of India court case records spanning the years 1990 to 2018. 1000 for testing and the remaining for training.NLTK was used to perform standard preprocessing methods. Afterward, a TF-IDF score was used to weight each word. The TF-IDF values of the words that make up each phrase are totaled and

standardized for sentence length. To construct their models, two techniques are employed: CaseSummerizer is an unsupervised approach, and LetSum and GraphicalModel are supervised approaches. To assess their application, both qualitative (WestLaw Gold Standard Summaries) and quantitative (ROUGE score) methods are used. They conclude that there is no one best way. While LetSum is the best way to depict the case's facts, GraphicalModel is a superior way to depict the statutes and precedents cited. Concurrently, the execution time is also a critical component in an online environment. The most appropriate assessment metric to gauge the caliber of domain-specific summaries may not necessarily be ROUGE scores. Creating a strong and sizable collection of gold-standard summaries for training supervised techniques, particularly neural models, is a significant upcoming challenge.

According to Xiao et al. (2021), legal activities are still difficult for PLMs to do since legal papers typically have thousands of tokens, which is far longer than what standard PLMs can handle. They suggest Lawformer, a pre-trained language model for interpreting Chinese legal long documents that are based on Longformer. Lawformer is assessed on a range of LegalAI tasks, such as question responding, retrieving related cases, legal reading comprehension, and decision prediction. The CAIL-Long dataset, comprising 1,099, 605 civil cases and 1, 129, 053 criminal cases, was utilised . Longformer is used in their model as a simple encoder. Global attention dilated sliding window attention, and sliding window attention are the three attention methods. As assessment metrics, they used the top-k Mean Average Precision (MAP), Precision (P@k), and Normalized Discounted Cumulative Gain (NDCG@k). The outcomes of the experiment show that their model can make significant progress on jobs requiring lengthy papers as inputs. Comparable outcomes can be obtained using L-RoBERTa and Lawformer. They declare that they will also investigate the generative legal pre-trained model

because practitioners in the real world of law must perform extensive and repetitive paper writing tasks.

The study presented by Holia, and Pandya (2023), identifies that conventional customer service methods such as FAQs are outdated and require a paradigm change that would prioritize personalized, context-aware, and responsive interactions. The authors propose a creative solution based on LangChain, a unique Large Language Model (LLM), to handle the changing dynamics of customer services in the digital age. The dataset utilized in this study was gathered using web scraping techniques, which collect publicly accessible data from a company's website, such as chat logs, product manuals, FAQs, and support forums. This scrapped data is used as background information to train the LLM. They proposed a framework, called "Sahaay," a cutting-edge system that incorporates customer support platforms with LangChain. The utilization of web scraping to gather data, vectorizing text using embeddings-HuggingFace Instruct Embeddings, and choosing Google's Flan T5 XXL language model for knowledge retrieval and answer creation are the main constituents. The framework is made to be adaptable to different organizations and sectors. By evaluating the answers to a range of queries, the authors compare the performance of three language models: Google's Flan T5 XXL, BASE, and SMALL. Their results show that the XXL model performs better inside the given framework. Though the results show that the XXL model is beneficial, improving the evaluation process could yield more reliable information. However, certain limitations, such as the data collection's dependency on web scraping and the need for more research into multimodal capabilities for information extraction from different file kinds are mentioned.

Saito et al. (2024) highlight the fact that a very crucial task of LLMs is to generate correct information which it does through self-supervised training, acquiring a variety of knowledge

from large-scale training datasets. However, the challenge lies in attempting to solve how these pre-trained LLMs would work or adjust automatically for question-answering tasks to different targeted domains or contexts without the need for costly annotated data. To handle this challenge, authors propose a novel approach of building a QA model specific to the target domain that can respond to target-specific queries of that particular domain in an unsupervised manner by using publicly available QA datasets, pre-trained LLM, and unlabeled documents from the target domain. Three different datasets are used- a synthetic biography dataset, a real-world dataset named Paper2023, and the News2023 dataset. The paper2023 dataset is a collection of abstracts and paper titles from various conferences, and the News2023 dataset was created by gathering news articles from the web. The authors mention that real-world datasets test the model's performance in realistic scenarios. The pre-trained LLM are fine-tuned on the QA dataset and unlabeled documents of a particular domain. However, the authors identify the limitations of the fine-tuned model's capacity to retrieve data from the middle or end of the papers is limited, probably because LLM training is auto-regressive. They discovered a method known as random token replacement to tackle this issue during the adaptation phase. The authors create evaluation questions based on the properties or data found in the papers and then evaluate how well the model can retrieve accurate responses. The model pre-trained with instruction tuning performs significantly better than a model trained only with self-supervision.

The study "Scaling Instruction-Fine Tuned Language Models" by Chung et al. (2022) focuses on expanding the model size, scaling the number of tasks, and checking the fine tune process by integrating chain-of-thought (CoT) data. The authors explore and discuss how instruction finetuning can improve and enhance the language models' performance, functionality, and generalization capacity. For this experiment, the authors used a wide range of datasets

incorporating over 60 original instruction tasks and different domain datasets covering 1.8k jobs. They introduced a new language model Flan with a scaled model size of 540 Billion-parameter, fine-tuned on a huge pre-trained model called PaLM and T5 which was trained on CoT data and 1.8K fine-tuning tasks with the need to improve the earlier instruction fine-tuning approach that faced performance degradation for CoT data. A variety of benchmarks such as RealToxicityPrompts, TyDiQA (Typology-Dense Question Answering), BBH (BoolQ, Balanced Holistic), and MMLU (Multitask Motivated Metrics for Longform Question Answering) including open-ended generation tasks are used for evaluation purposes. The results reported are Flan-PaLM reaching state-of-the-art outcomes like 75.2% on the five-shot MMLU task. Authors mention the potential limitation of performance of other task types other than CoT tasks.

Comparison Among Relevant Research Papers

An extensive literature review is performed to understand different datasets and methodologies used to get a grasp of the Large Language Model operation (LLM) operation. Miller (2019) introduced the architecture to summarize the lectures where BERT embedding and K-Means clustering are used to efficiently summarize the lectures. Jain et al. (2020) concentrated on the extraction of information from the document and also introduced a new dataset called SCIREX that is suitable for extracting relationships from scientific articles and proposed a neural model for understanding the document content. Adams et al. (2024) focus majorly on using LLM for interpreting medical records to determine how the LLM performs during the process of text retrieval and data extraction from different clinical records. Saad-Falcon et al. (2023) proposed PDFTriage to perform question-answering of the documents, where comprehensive and accurate answers are provided, and Medeiros et al. (2023) discusses the development of the chat-bot application based on LLM that uses the vehicle’s manual and provide the answer. Each of these

studies focuses majorly on the specific application of the LLM that shows the effectiveness of the LLM in different domains. Transitioning toward the studies that talk about the application in the legal world, Chen et al. (2023) introduced a new dataset to perform effective legal-based question-answering known as EQUALS. Louis et al. (2023) proposed another dataset named LLeQA to overcome the issues in the existing datasets related to laws by creating a pipeline that provides a long response to any user queries related to statute laws. Bhattacharya et al. (2019) addressed the issues of summarizing large documents related to legal laws. Analyzing all the studies mentioned here and others explored above helped to provide an understanding of the different concepts related to LLM and gave an overview of the existing research on processing legal law documents and providing answers to queries using LLMs.

Data and Project Management Plan

Data Management Plan

The data management plan describes the steps involved in collecting and processing data to obtain useful information. Along with collection and management, storage and usage mechanisms are discussed. Individual team members taking on roles and performing tasks are detailed as well.

Data Collection Approach

The dataset is retrieved by leveraging Colab Pro, a cloud-based collaborative coding platform, to scrape PDFs related to legal laws from the United States Code and Westlaw. The USC is administered by the Office of the Law Revision Counsel of the United States House of Representatives. The organization creates and releases the United States Code, which outlines the country's general and permanent laws. USC has 54 titles and has had continuous updates

since its initial publication in 1926, which is nearly a century. Therefore, this makes for a valuable asset to carry out the project.

Westlaw is owned and operated by Thomson Reuters, a multinational media and information company. It was founded in 1975 and provides access to over 40,000 databases (Wikipedia contributors, 2023). It includes case law, legislation, administrative regulations, publications, public documents, and legal magazines. Commerce and trade-related data are covered by Title 15 which encompasses a wide range of topics such as company regulation, consumer protection, competition, and trade practices.

External libraries, such as pdfplumber, are installed to carry out PDF parsing tasks. pdfplumber helps to retrieve text and metadata from PDF documents, making it easier to get relevant legal data. The main task was to set up the automated pipeline to extract data from different sources and to process them efficiently to ensure that the extracted data is suitable for performing further actions. The data is extracted from all the chapters present in all the titles, with a focus on content available as of 2023. This approach aims to ensure that any changes or amendments to the regulations are right away reflected in the codebase.

The legal case documents like the ones that involve the Capital One data breach, access to confidential details of Sea Mar Community Health, etc are extracted manually by checking their relevance with the decided problem.

The policies provided by different companies and terms of use and service issued by the companies related to their various services are extracted from the GitHub repositories and some of them are manually downloaded by navigating to the different company's websites. The extracted policies are processed using the pdf2image library which converts each page in the PDF document into the form of an image and using the technology of Optical Character

Recognition (OCR) extracts content from it. The aim is to ensure that important and relevant information is not lost.

Data Management Methods

The integrity and accessibility of the legal datasets are guaranteed by using effective data management techniques. Using version control systems, like Git, makes it possible to monitor changes made to the models and datasets, which promotes repeatability and collaboration. Cloud-based storage options like Google Drive are used to provide safe and scalable storage for both raw PDF data and processed embeddings. Data reliability is enhanced by routine data backups and audits, which protect data from possible loss or corruption. Using data governance techniques, such as encryption and access restrictions, also guarantees adherence to privacy and legal requirements, improving overall security and responsible handling of sensitive legal data across the project lifetime. Table 3 mentions roles and assignees.

Table 3

Roles and Assignees Used for Data Management

Roles	Assignee	Tools
Data Collection	Yuti, Sohail	Python Script
Data Cleaning	Namratha, Swetha	Colab Pro
Data Processing	Sourab, Sohail	Colab Pro
Data Storage and Sharing	Swetha, Namratha	Astra DB
Data Management & Data Exploration	Yuti, Sourab	Jira, Python
Data Modeling	Namratha, Swetha, Sohail	Python Script
Model Evaluation	Sourab, Yuti	Python Script
User Interface	Sohail, Swetha	Python Script

Note. The above table shows the roles that are assigned to each team member. Original from Team 1.

Data Storage Methods

To deal with PDF content and legal papers, a vector database turns out to be a wise decision for effective data storage. Because vector databases like Astra DB can handle high-dimensional data representations well, they are used to store embeddings that are taken out of PDFs. With the use of methods like document embeddings or word embeddings, the PDF content is transformed into numerical vectors for fast and precise retrieval. Using a vector database guarantees scalability and quick search speed, and it makes it easier to conduct semantic similarity queries—which are essential for quickly obtaining pertinent legal information. This storage strategy emphasizes the value of content-based retrieval and analysis in the context of legal documents, which is in line with the project's needs.

Data Usage Mechanisms

The Large Language Models (LLMs) are effectively trained using the extracted embeddings from the PDFs, allowing them to understand and produce information related to IT laws and regulations, cases, and policies. The models are fine-tuned by modifying their parameters and providing the question-answer pairs generated from the legal documents to improve their ability to generate meaningful answers to the queries and comprehension of complex legal terminology and contextual nuances. Furthermore, the vector database proved to be useful for retrieving content in real-time, which enables the chatbot application to quickly retrieve and deliver pertinent legal information while interacting with users. Strong data pipelines and version control systems guarantee that the model is improved over time, which follows the changing legal regulations, and policies and promotes the continued development of the AI-powered legal assistant.

Project Development Methodology

CRISP-DM Methodology

CRISP-DM intelligent system development cycle is being followed in the execution of this project. Key steps in this cycle include understanding the problem, gathering and processing data, choosing LLMs, fine-tuning them, evaluating them, and deploying them. Effective management of these scheduled development processes and activities is required. The six phases of project development:

Planned Project Development Processes and Activities

Problem Definition. Establishing the primary aims and objectives of the project was the initial stage of the procedure. The project's goal is to tackle the intricate legal frameworks that protect online privacy, such as commerce legislation, Federal Trade Commission (FTC)

guidelines, protection of intellectual property rights, etc. The project also focuses on getting a thorough understanding of the legal cases, policies followed by the companies, and terms of use and service defined by the tech companies for their various services and responding to user queries. The project's objective is to use large language models (LLMs) to create an AI-powered legal assistant. With the use of this application, anyone with no prior legal experience would be able to quickly explore and gain an understanding of legal literature, improving compliance and comprehension of intricate legal frameworks. This will also help the person who recently filed a case against tech companies regarding violation of tech-related laws can use previous legal cases along the same lines to get an idea about the next set of operations that could be performed to have higher chances of success.

Data Collection. Our approach includes gathering data from multiple websites to improve the Large Language Model (LLM) with a specialized understanding of technology laws and regulations. This data which includes policies, historical legal cases, and terms of service, is subsequently placed into a designated folder within our shared drive. Through implementing automated code, we ensure that whenever a new PDF document is added to the drive, our system automatically incorporates it and executes all necessary steps to integrate its content into the LLMs. The chosen datasets included a wide variety of legal papers pertaining to technological legislation, such as trade regulation guidelines, Federal Trade Commission (FTC) standards, and other pertinent resources; policies followed by different companies like Business Conduct Policy by Apple; historic cases filed by people or organizations against tech companies related to the violation of laws related to technology; and terms of use and service defined by the companies such as Whatsapp that is needed to be followed by the users to avoid any compliance issues. For the LLMs to be properly trained to comprehend and produce information pertaining to legal

requirements, the quality and scope of the data is essential. A vector database is used to store the data.

Data Analysis. Conducting exploratory data analysis (EDA) on PDF data related to tech legislation is essential for understanding the properties of the dataset and providing guidance for Large Language Model (LLM) optimization. Basic dataset information including the number of documents, pages per document, sentence length, each word length, and special characters in the textual content are extracted as part of the EDA process. This gave a general idea of the quantity and complexity of the dataset. An analysis is conducted on the most commonly used terms and phrases found in legal papers.

Model Selection and Fine-Tuning. Pre-training goals, resource availability, and model architecture are some of the considerations that go into choosing the right Large Language Models (LLMs) for a project. Models with strong context retention, generation skills, and legal language understanding—such as Open AI, Gemini, Llama, and Mistral—are frequently employed. The LLMs are fine-tuned in order to be customized to the unique domain of tech laws, and regulations; policies; terms of use and service; and historically filed legal cases. This allows the model to comply with the complexities of legal structure, identify important entities, and understand complex legal language. The LLMs are adjusted to achieve greater precision and applicability, guaranteeing their efficacy as a specialist instrument for negotiating intricate legal frameworks in the legislative branch.

Model Evaluation. Analyzing the chat application's success requires a thorough process that incorporates both qualitative and quantitative evaluations. User feedback and human evaluation are two qualitative methods for evaluating user experience and interaction. Key qualitative indicators include evaluating the satisfaction of the users with the generated response,

and whether the response delivers correct and contextually relevant legal information. Metrics related to the legal environment, such as F1 score, Perplexity, and Open AI Evals are used to quantitatively assess how accurate the chatbot is in responding. Furthermore, monitoring response times, user involvement, and the number of successful interactions over time provide quantitative insights into how well the AI-powered legal assistant meets project goals and how efficient and effective it is.

Documentation. Transparency, reproducibility, and the continuous development of the LLM-powered chat application depend on thorough documentation of the entire process. A thorough documentation plan contains thorough records of the sources used to obtain the data, the preprocessing procedures, the model architecture selections, the hyperparameter setups, and the training procedures. Knowledge transmission requires detailed instructions on how to install the chat program, set up the platform, and update the model. To ensure regulatory compliance, the paperwork highlights privacy precautions, security measures, and legal concerns that are followed throughout the project. Frequent updates to the documentation support future legal tech attempts and help ensure the project's long-term viability. These updates also include any revisions or enhancements made to the model.

Project Organization Plan

The organization of the plan for the project is considered one of the most important processes as it sets the layout for the tasks that are performed or will be performed in the project. The work breakdown structure, also known as WBS, is used in this project to show the phases involved, what tasks are going to be performed, and what deliverables will be provided. The CRISP-DM methodology is being adopted while designing the WBS, which consists of six phases where each of the phases represents the different set of tasks that will be performed to

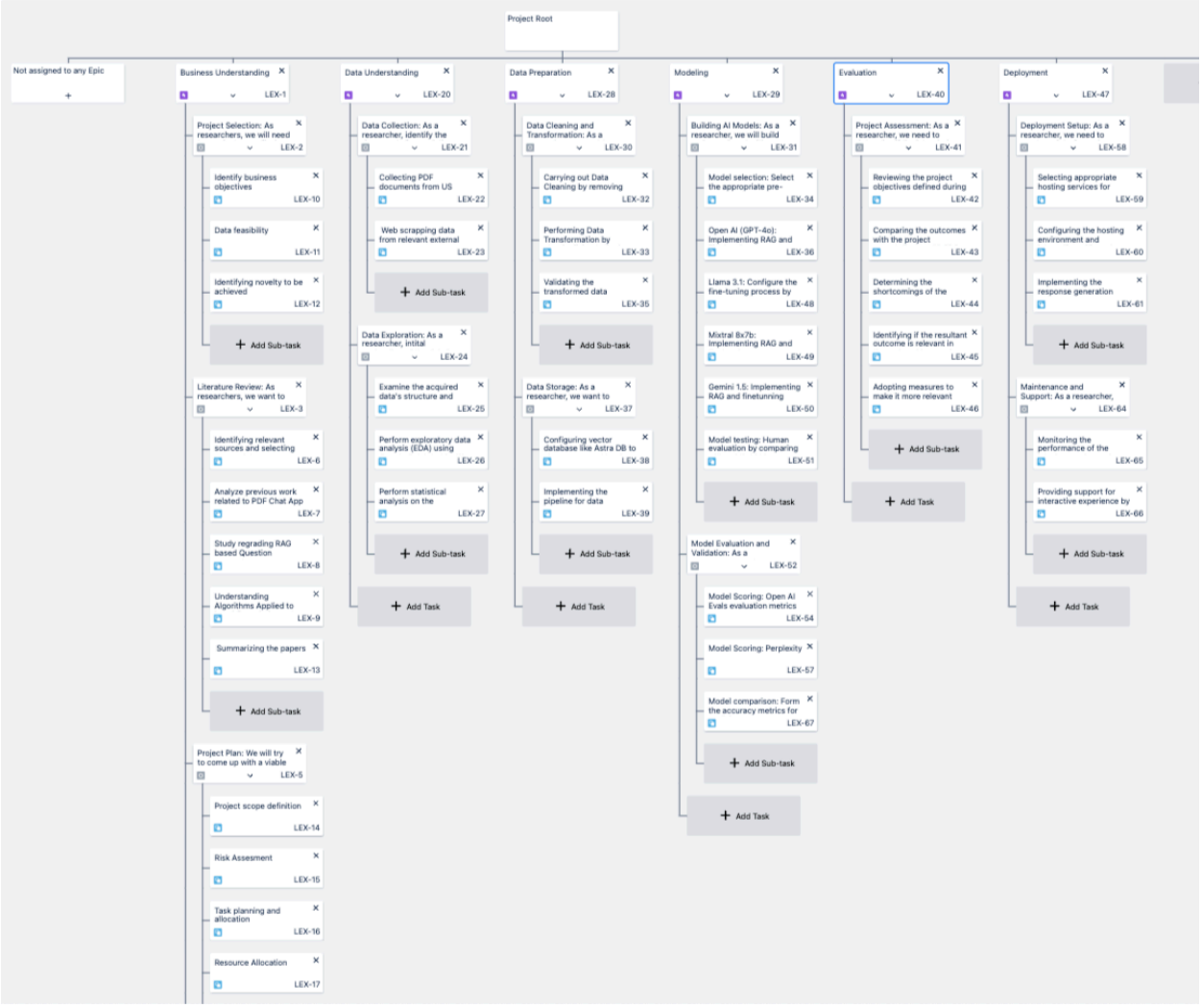
progress towards the goal of completing the project with necessary deliverables.

The first phase is business understanding which consists of four user stories. The project was initiated by determining the feasibility of the problem statement and deciding whether to go ahead with this or think of a different path. Also, thinking of the novelty that could be achieved in it. Extensive research of the existing papers was performed where the aim was to determine the papers that talk about the implementation of the chat applications where either the PDFs, books, or articles are being used to fine-tune different LLM models. Some papers are also explored to get a better understanding of how RAG-based question-answering works. There are a few papers that talk about the usage of LLMs to answer queries related to legal laws which served as a useful resource. A proper and quite realistic project plan was developed to keep track of the deadlines. The second is the data understanding phase which is divided into two user stories where the data (PDF documents) is collected from various sources like the US Code and Westlaw websites; polices adopted by different companies, information about the historic legal cases filed against the violation of tech related laws, and terms of use and service provided by companies to use their various services are downloaded manually and performing initial data exploration to determine if the structure of the document is suitable, and get brief insights about the textual information by either creating various visualizations or performing statistical analysis like word frequency. In the data preparation phase, various useful techniques are incorporated to preprocess the data to make it suitable for modeling and storing it in appropriate locations. Tasks like data cleaning and transformation are performed to prepare the data for the model and the processed data is stored in a vector database named Astra DB. In the modeling phase, the initial task was to select the appropriate models for the problem statement which is followed by training and testing the models. The models will be evaluated on different evaluation metrics like

Perplexity, Open AI Evals, and human evaluation; and further compared against the benchmark scores. In the evaluation phase, the assessment of the project will be performed to determine if the outcomes obtained are in line with the expectations if all the objectives are properly met, and if there are any shortcomings then what are the possible methods to resolve them. The final phase will be deployment where the chatbot application will be deployed on the interface that will allow the user to provide queries regarding the laws and LLM will provide the response. There will be constant monitoring of the interface to ensure its proper operation and support will be provided in case of need. A detailed illustration of WBS is provided in Figure 2.

Figure 2

Work Breakdown Structure (WBS) of the Project



Note. Complete work breakdown structure of the project along with all the phases involved. Original from Team 1.

Project Resource Requirement and Plan

Hardware Requirements

Artificial Intelligence is still an ever-evolving field and the latest development in the field of Generative modeling is groundbreaking but the Large Language Models are complex and they require the usage of heavy computational resources. They contain many complex algorithms that

handle multiple sections of data at the same time which could increase processing time and require more resources. If enough resources are not available then it may lead to an increase in the training time of the models which could ultimately cause a delay in the schedule and requirement of resources for a longer time which increases the cost.

To ensure that various computations that require the usage of LLMs can be performed efficiently and quickly there is a need for proper devices with high hardware specifications like devices with powerful CPUs and GPUs, and large enough memory to allow smooth operation of various tasks like model training to improve the accuracy and achieving intended goal in more proper extent. Table 4 illustrates the hardware requirements for different processes in the project.

Table 4

Hardware Requirements of the Project

Hardware	Memory (RAM)	Configuration	Purpose
14 core CPU and 2560 CUDA Core GPU	12 GB and 8 GB dedicated	6 performance core 8 efficient cores	For different LLM frameworks, performing development and ensuring RAG implementation, and fine-tuning of LLM models
16 Core GPU	128 GB		For fine tuning LLM models
T4 GPU	16 GB		For data exploration and preparation

Note. List of different hardware specifications that will be needed for fine-tuning LLMs. Original from Team 1.

Software Requirements

To perform a lot of different operations like data cleaning, transformation, and model training in the project various software libraries and packages are required and they are needed to be considered along with the hardware requirements. It is quite crucial that the software packages that are used are compatible with other packages to ensure smooth processing and ensure effective performance. Different software libraries, the methods, and how they are used are mentioned in Table 5.

Table 5

Packages and Libraries Used in the Project

	Library	**Method**	**Purpose**
Operating System	os	environ	Environment variable that provides a token that can be used to permit API requests
Streamlit	streamlit	write	To provide details like textual information, model training or fine-tuning process, etc.
		file_uploader	To upload PDF documents that are used to train or fine-tune LLM models
Python PDF version 2	PyPDF2	PdfReader	To read and extract content in the PDF documents and to also extract metadata information
LangChain	langchain.llms	OpenAI	To access and fine-tune OpenAI's LLM model

Library	Method	Purpose
langchain.text_splitter	CharacterTextSplitter	To split the textual content in the PDF documents into individual characters or perform effective tokenization
langchain.embeddings.openai	OpenAIEmbeddings	To access the embeddings created for the textual content that are derived from the language model of OpenAI
langchain.vectorstores	FAISS	To manage, store, and query the vector form of the content present in PDF documents that involves tasks like retrieval of documents, performing similarity analysis ,and vector indexing and querying
langchain.chains	LLMChain	Serves as a pivotal framework to ensure effective and proper interactions between different components of LLM performed effectively.
langchain.prompts	PromptTemplate	To generate structured prompts that help to guide the way the LLM model performs for a specific task and generate appropriate responses to the user queries.

	Library	Method	Project
Astra DB	astra-python	Client	To initiate the connection between LLM and Astra DB; and to store data and perform queries
Cassandra	cassandra.auth	PlainTextAuthProvider	To provides authentication for Cassandra cluster
	cassandra.cluster	Cluster	To represents a Cassandra cluster
Llama Index	llama_index.core	ServiceContext	Represents the context for a service in the Llama Index core
	llama_index.core	set_global_service_context	Sets the global service context for the Llama Index core
	llama_index.core	VectorStoreIndex	Represents an index for vector stores in the Llama Index core
	llama_index.core	SimpleDirectoryReader	To reads a directory in a simple manner
	llama_index.core	StorageContext	To represent the context for storage in the Llama Index core
	llama_index.vector_stores.cassandra	CassandraVectorStore	Represents a vector store in Cassandra
	llama_index.embeddings.gradient	GradientEmbedding()	Create a gradient embedding for the LlamaIndex service

	Library	Method	Project
	llama_index.llms.gradient	GradientBaseModelLLM()	Create a gradient-based model LLM for the LlamaIndex service

Note. List of libraries/packages along with some of the methods that are/will be used. Original from Team 1.

Databases play a pivotal role in projects that involve large amounts of data and it is one of the most important requirements along with hardware and software. Table 6 provides information about the database used to store data with its purpose and the version used. Table 7 lists the tools and Licenses that are used during the course of this project.

Table 6

Database Used in the Project

Database Name	Purpose	Version
Astra DB	Cloud database based on Apache Cassandra and provides vector DB for LLMs	2023

Note. Database that will be used to store our vector embeddings. Original from Team 1.

Table 7

Tools and Licenses

Tools Name	Purposes	License
Google Colab	Cloud-based environment for execution of RAG framework.	Proprietary
GitHub	Web-based software development system	Various
Excel	Spreadsheet software	Proprietary

Tools Name	Purposes	License
Google Docs	Web-based word processing	Proprietary
Google Meet	Conferencing software	Proprietary
JIRA	Web-based project management tool	Proprietary
Grammarly Premium	Web-based tool for checking grammatical errors	Proprietary

Note. List of Tools and Licenses required. Original from Team 1.

Project Cost and Justification

Table 8 provides a breakdown of the project's resource costs and justification. Each expense and the costs related to the resources needed to finish the project are explained in this table.

Table 8

Project Resources Cost and Justification

Resource	Justification	Duration(Months)	Cost
Hardware Cost	Required hardware to perform the research	4	$1600
DataStax Cloud Database	Database used to store the data	4	$150
Grammarly Premium	Web-based tool for checking grammatical errors	4	$120

Note. List of the resource's cost and its justification. Original from Team 1.

Project Schedule

Gantt Chart

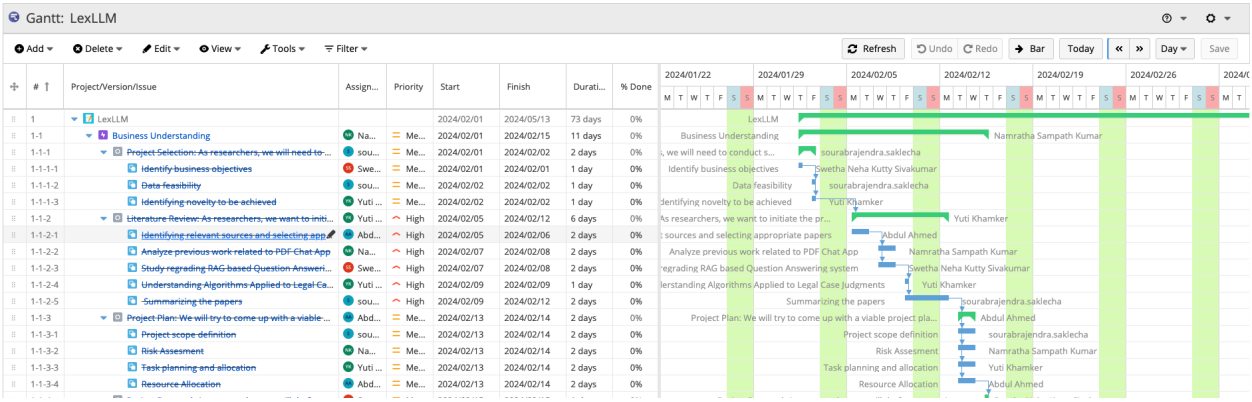
Gantt chart is a visual representation for scheduling the project that helps the team plan with keeping the track of the allocated tasks and their status at each phase along with identifying dependencies and relationship between each task and maintaining the transparency within the team and stakeholders. For this project, Gantt Chart is created in Jira using third-party plugins or apps called WBS Gantt Chart.

The Gantt chart adheres to the waterfall concept, which states that each project phase starts after the preceding ones and that there is no overlap between them. Figure 3 provided below shows the initial and the first phase of the project which is business understanding. The goal of this phase is to first select an appropriate project topic by identifying business objectives, checking data feasibility along with a novelty approach that can be achieved. Main task is to determine the specific domain by conducting extensive literature review by identifying relevant papers on large language models, related work on PDF Chat Applications, RAG based Q&A system and algorithms applied. To get an final overview of the methodology proposed, the dataset and evaluation metrics used in each paper, a summary of all the papers is documented. This is a crucial phase as it provides clarity on existing approaches and highlights any limitations or flaws. This phase is given high priority. Next task is to plan the project which includes defining the project scope, potential risk involved, planning tasks for the team, and allocation of the tasks and required resources. Every team member is assigned some tasks that are to be completed. Additionally, it was made sure that each team member had about the same amount of hours, ensuring that the workload was divided equally. To maintain the granularity in tasks, no

tasks exceeded more than 2 days limit. Parallel tasking is implemented. Figure 3 illustrates the different tasks involved in the business understanding phase.

Figure 3

Gantt Chart for Business understanding Phase

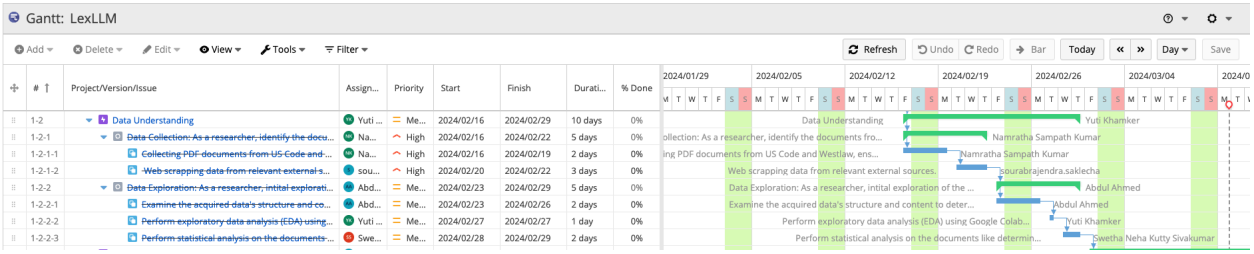


Note. Business understanding phase and all the tasks under this phase. Original from Team 1.

Figure 4 depicts the phase 2 of the project which is data understanding and the aim of this phase is to have a thorough understanding or grasp of the data and its quality. This phase starts with data collection in which it involves a very important task like identifying the data source as it helps in determining the quality of the data and then after collecting PDFs from US Code and Westlaw, policies adopted by companies, terms of use and services released by tech companies, and past historic legal cases their relevance with the project is determined. Once the data is collected, the next task is to carry out exploratory data analysis to examine the data structure, quality and suitability for training. EDA was performed on the documents to determine work frequency, richness of vocabulary, distribution of word and sentence lengths etc in Google Colab to identify patterns and insights. Same as in the first phase, each task is assigned to all the team members and all the tasks are completed.

Figure 4

Gantt Chart for Data understanding Phase

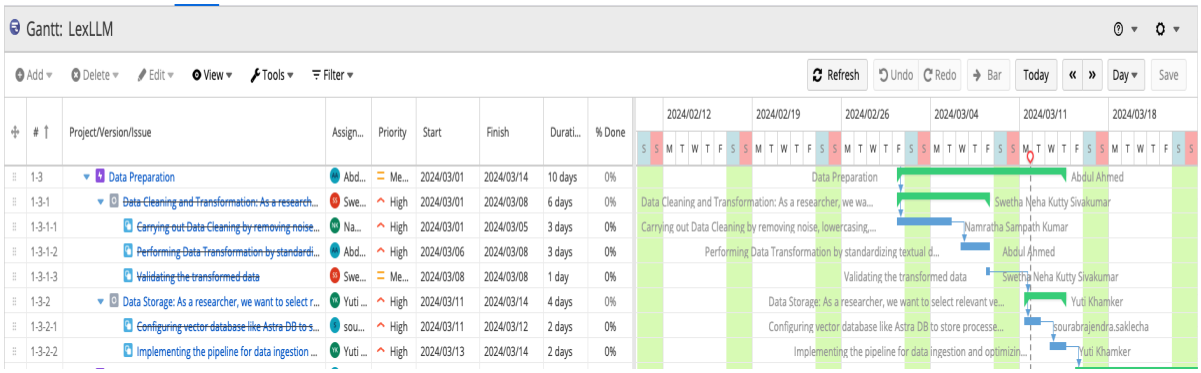


Note. Data understanding phase and all the tasks under this phase.Original from Team 1.

Figure 5 shows the data preparation phase which is focused on preparing the raw data for further analysis. Phase 3 starts with a thorough data cleaning to the pre-loaded data on google colab. Before directly utilizing the data, it is important to clean the data and that's why data cleaning is given a high priority in the chart. Cleaning process involves removing noise and special characters, lowercasing and tokenization. This aids in providing data free from all the quality issues. After this task, data transformation is done by standardizing the textual data and applying lemmatization. Under data preparation, an important task after cleaning is to configure data storage to select relevant vector databases like AstraDB to store processed legal textual information. Once the AstraDB is configured and vector embedding is stored in this database, the next task is to implement the pipeline for data ingestion and optimizing storage. All the tasks from data cleaning, and transformation to storage are completed by the assignees within the timeframe in this phase. The status of this phase is completed.

Figure 5

Gantt Chart for Data Preparation Phase

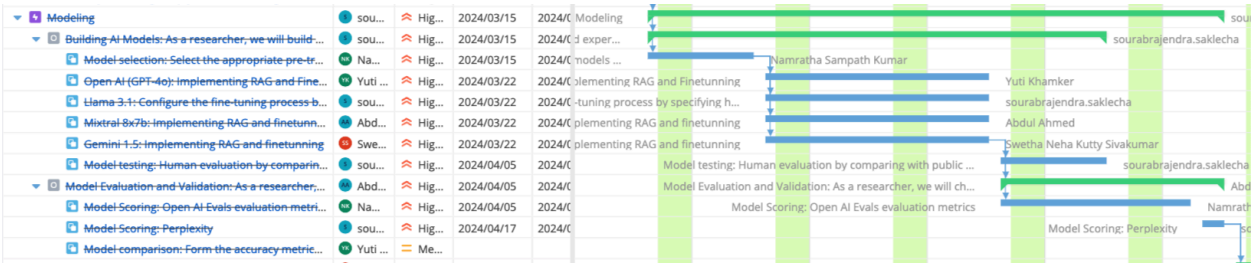


Note. Data preparation phase and all the tasks under this phase.Original from Team 1.

Figure 6 shows the tasks and dependencies in phase 4 which is modeling. The objective of this phase is to experiment and evaluate the Large Language Model using Langchain capabilities that effectively answers the user queries. This phase initiates by doing research on different Language models that could be used as PDF Chat Application, selecting some appropriate pre-trained models like OpenAI (GPT-4o), Llama 3.1, Mixtral 8x7B and Gemini 1.5 as a base model for this project. Other tasks like splitting the dataset into training and testing data to train the above-mentioned models and testing the accuracy of these models are carried out. One of the important aspects of this project is to utilize these models with customized hyperparameters. Configuring the fine-tuning process by specifying the hyperparameters and training one of the models and compare it with the other four RAG models followed by model testing with different public LLMs. The next task involves evaluating the accuracy performance of each model against different evaluation metrics like Perplexity, Open AI Evals, and Human Evaluation. The final task would be to have a model comparison by forming accuracy metrics for model vs score. In this phase, all the tasks were given the highest priority. The status of this phase is “to-do”.

Figure 6

Gantt Chart for Modeling Phase

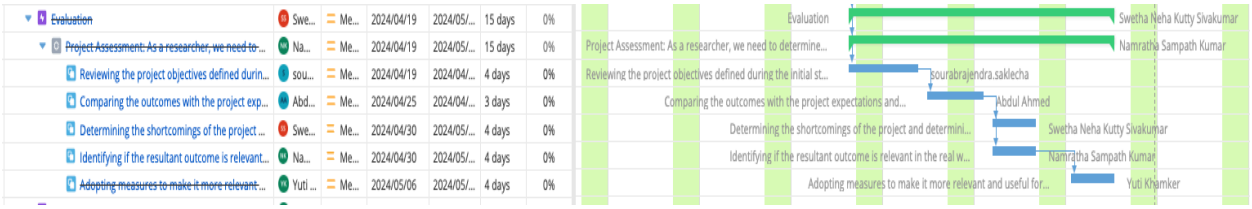


Note. Modeling phase and all the tasks under this phase.Original from Team 1.

Phase 5 is about the evaluation of the project. Figure 7 shown below investigates the outcomes and determines if these outcomes are close to the intended goals and whether they met the expectations. The task in this phase involves assessing the project by reviewing the project objectives during the initial stage of the project. Comparing the outcomes and determining the shortcomings of the project and devising a plan on how to overcome them. Identifying the resultant outcome is relevant in real-world scenarios and adopting measures to make it more relevant for the intended audience. The status of this phase is “to-do”.

Figure 7

Gantt Chart for Evaluation Phase



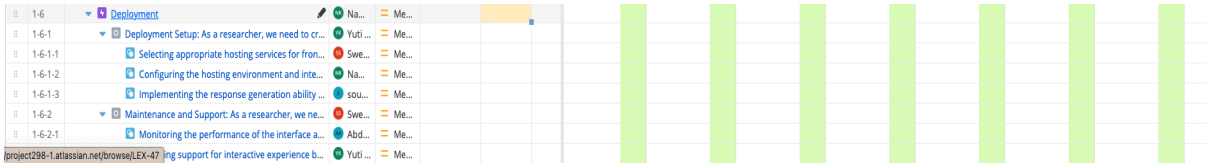
Note. Evaluation phase and all the tasks under this phase.Original from Team 1.

Figure 8 shown below is the final phase of the project- The deployment phase. In this phase, the LexLLM chatbot application will be built that allows the processing of queries from the user and determining methods to provide maintenance and support. Multiple tasks will be

incorporated in this phase which involves creating an interface by selecting appropriate hosting services for front-end and backend and configuring the hosting environment and integrating it with the vector database and implementing the response generation ability on the interface. Currently the status of this phase is pending as this phase can only be completed once all the above phases are marked as done. Dates for this phase have not been provided purposely.

Figure 8

Gantt Chart for Deployment Phase



Note. Deployment phase and all the tasks under this phase.Original from Team 1.

PERT Chart

The PERT chart also known as the Project Evaluation and Review Technique (Bagshaw, 2021) chart is a commonly used project management tool that is used to keep track of the different tasks that need to be performed to achieve the project's goals. It is used to outline projects with high complexity as it provides the breakdown of the project in the form of tasks that need to be performed in a sequential manner in which they happen. It provides information about the dependencies involved between different tasks, the number of days required to complete that task, and the start and end dates of them. It is a pictorial representation where the tasks are considered nodes and the arrows with direction going from one task toward the other represent the dependency of that task over the one having the arrow (>) symbol on it. This proves helpful in providing an overview idea about the workflow of the project.

Task-based approach is selected to create a PERT chart in this project. The main aim is to determine the tasks that are involved in the project and to identify the critical path by using a

technique called Critical Path Analysis (CPA). The critical path is useful as it helps to understand the tasks that need to be performed at any cost and to identify the ones that could be avoided to achieve the goal quicker than anticipated. It tries to determine the path that involves all the important tasks and that can be achieved within the minimum time to finish the project. This path signifies that the tasks that are present along this path cannot be delayed than their mentioned end dates because of the dependencies of the next task on this one, so if one of the tasks gets delayed the complete project will shift to the many days that it gets delayed. This could lead to delays in meeting the deadlines of different deliverables. Non-critical tasks always have the option to get right back on track even after they get delayed and their delay doesn't affect the deadlines.

Figure 9 shows the PERT chart of the project where the information about the tasks, dependencies between tasks, and the critical path identified is provided. Tasks are represented as rectangular boxes that contain information like the task name, the unique number representing the task's number, the start and end dates of that task, and the number of days assigned to finish that task. The dependencies show that the next task will start only after the previous one finishes. The critical path contains the critical tasks that are represented with red color boxes with the dependencies represented as red directional arrows whereas non-critical tasks are represented as blue rectangular boxes with their dependencies represented as black arrows.

In this project, determining the critical path is useful to ensure that all the important tasks are performed within the assigned time so that the project's intended goal can be achieved effectively. The first critical task is initialing the project and determining its relevancy and what novelty could be achieved, followed by getting an understanding of the domain of interest and identifying the business problem of creating chatbot applications where the LLM models are

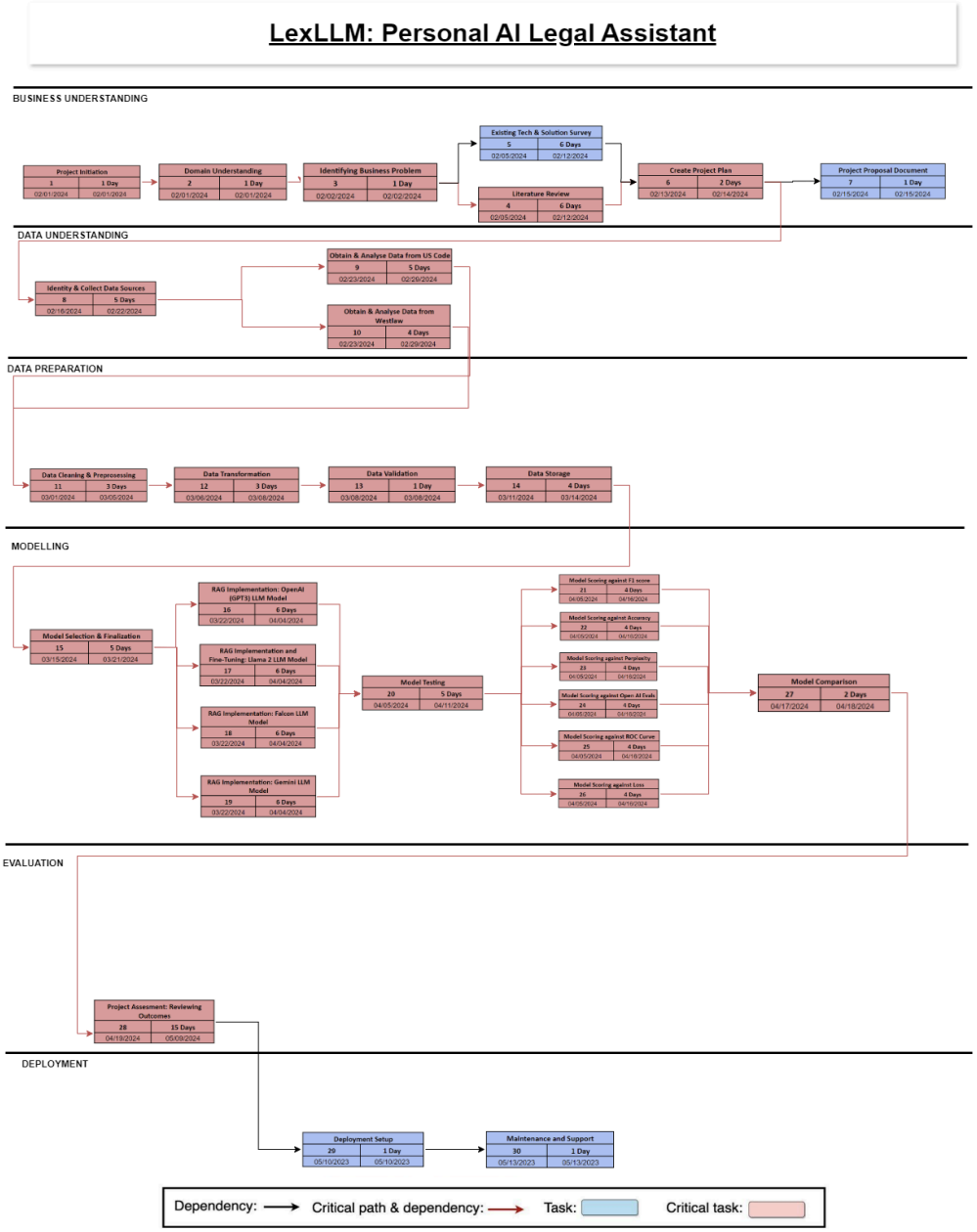
trained on various legal laws, policies, terms of use and service and past legal cases to answer user queries, and performing extensive literature and technology survey to get a better understanding of the different LLM models and how they work, how LLMs could be fine-tuned using legal laws PDF documents, how the response is provided to various queries related to legal field provided by the users, and what evaluation metrics can be used to evaluate model; devising a realistic project plan; identifying and collecting data from various sources.

The critical path continued with performing tasks like data cleaning and transformation to prepare data for the modeling, followed by performing data validation to check the appropriateness of the processed data, and data storage to store the data in the vector database.

Then model selection to determine useful models for the defined business problem, creating RAG based application using four selected models using the above-provided datasets that will be extracted from the vector database and also performing fine tuning of some of the models to generate more appropriate legal related response, testing the models by carrying out the human evaluation, and evaluating the models against various selected evaluation metrics to determine best-performing model against benchmarks. Some of the tasks that are critical like model RAG based application, fine-tuning and model evaluation or scoring against various metrics are/will be performed simultaneously by different team members to ensure efficiency.

Figure 9

PERT Chart of LexLLM: Personal AI Legal Assistant



Note. Complete task-oriented PERT chart that contains critical path and other non-critical tasks. Original from Team 1.

Data Engineering

Data Process

The project's goal is to create a chat application called LexLLM that can respond to user inquiries while managing the intricate details of the laws protecting children's online privacy. This AI-powered legal assistant is built by leveraging large language models (LLMs) that are trained on Legal documents. To accomplish the above law data consisting of multiple PDF documents is collected from the United States Code (USC), some of the policies adopted by companies like Apple or Google, legal laws filed against tech companies for violation of one or more technology-related laws, and terms of use and service provided by the companies like Whatsapp that is needed to be followed by the user to ensure compliance. The collected data undergoes extensive preprocessing steps involving data cleaning to handle inconsistency in the data. The data transformation phase regularizes the data that further is converted into question-answer pairs. The question-answer pairs are converted into word embeddings that are passed to the language models for training, validation, and testing. In this research, OpenAI GPT4o, Mixtral 8x7b, Llama 3.1, and Gemini 1.5 are utilized for the model development phase. The language models are further evaluated using metrics such as Perplexity, OpenAI Evals, and Human Evaluation. The embedding from the language models is stored in AstraDB. When the user queries, the user query embedding is passed to AstraDB where the similarity search between the user query and existing embeddings is evaluated to retrieve the relevant documents for the user query. The language model generates the response based on the relevant documents.

Exploratory Data Analysis (EDA) is performed on the collected data which helps in understanding the overall structure, content, and quality of the text data. It helps in understanding the total words, special characters, word length, sentence length, and POS tagging analysis to

understand linguistic trends. Analyzing the data exploration led to the need for the data cleaning phase.

The next step after EDA is the data pre-processing phase where the data is cleaned by handling the special characters, lowercasing the text to enhance the consistency of the data, removing stop words to discard the data with low semantic value, and pronoun removal to eliminate the pronouns using a filtered list of filtered words.

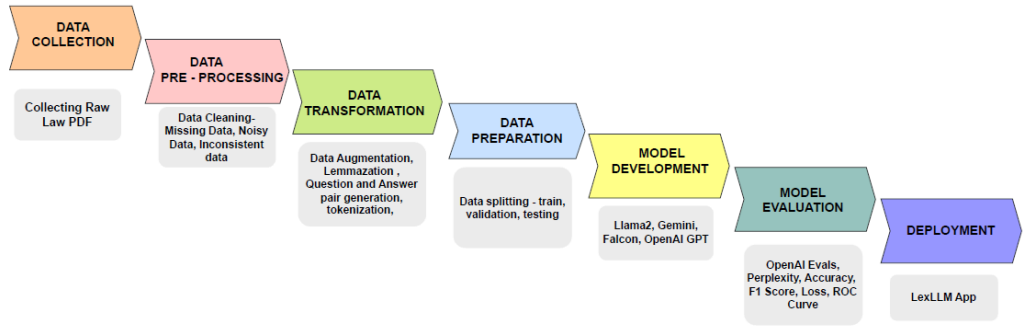
The cleaned data is transformed into embedding by a series of steps starting with data augmentation. In the data augmentation, the text data undergoes random synonym replacement, word swapping, and word deleting to enhance the model generalization. The transformed data is then split into chunks to generate question-answer pairs. The generated question-answer pair is then tokenized first followed by vectorization which is further split into training, validation, and test datasets for model building. The training data consists of 80% of the data, the validation set consists of 10% and the remaining 10% of the data is used for testing. Figure 10 below shows the data process workflow.

Data Quality Checks

As part of data quality checks of PDF files, every document is meticulously examined to confirm their accuracy, completeness, and consistency. Using the PyPDF2 module and Python scripting, the text content of each PDF is retrieved and analyzed methodically. This analysis consists of checking for missing or duplicate data, forecasting the distribution of special characters, validating data types, and identifying outliers. Each PDF is thoroughly validated to adhere to set standards. These data quality checks locate and fix any errors in advance, thereby improving the dataset's reliability and consistency.

Figure 10

Data Process Workflow



Note. The figure above shows the overall Data Process Workflow. Original from Team 1.

Data Collection

Data Source and Parameters

The data for this study was gathered from legal proceedings listed in the United States Code (USC). Other data such as Whatsapp Terms of Service, Qualcomm's Terms of use, Google's Acceptable Use of Policy, the Business Conduct Policy of Adobe, and the Compliance Policies of Apple were taken from the official website of the company which were downloaded manually. A few lawsuit cases regarding data breaches were filed by Hagens Berman. Some of the most relevant data are from the following chapters:

Chapter 2: Federal Trade Commission

Chapter 5: Federal Trade Commission's Trade Regulation Rules

Chapter 7: National Institute for Standards and Technology

Chapter 47: Consumer Product Safety

Chapter 91: The Children's Online Privacy Protection Act (COPPA).

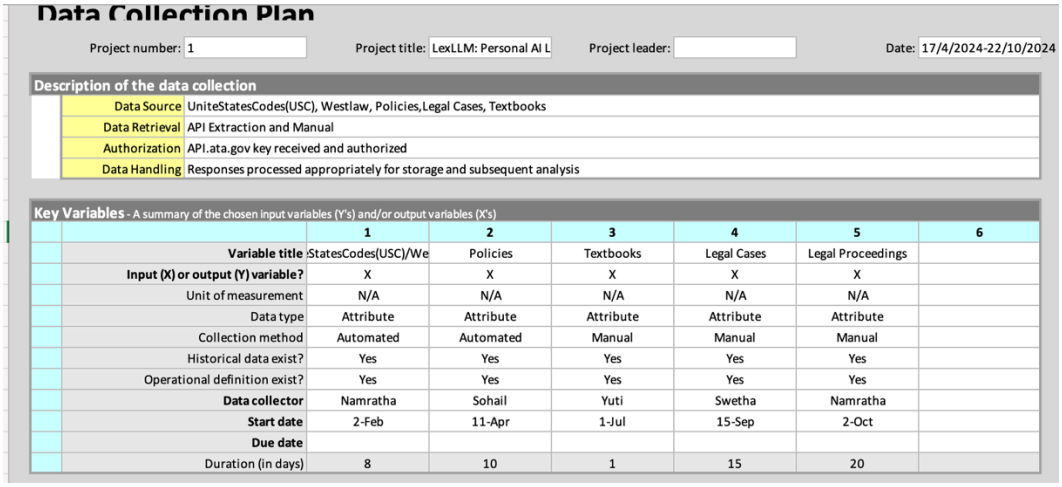
Chapter 107: Protection of Intellectual Property Rights.

The GovInfo API offers various services for accessing government material and metadata, making it an invaluable resource for developers and webmasters. An API.data.gov key is received to use the GovInfo API. This key acts as the authentication token, allowing you to send API queries safely. After signing up and receiving your API key, the next step is to authorize it.

Figure 11 illustrates the data collection plan. All the PDFs have the same set of features since the data is fetched through a single collection technique.

Figure 11

Data Collection Plan



Data Collection Plan

Project number: 1 | Project title: LexLLM: Personal AI L | Project leader: | Date: 17/4/2024-22/10/2024

Description of the data collection

Data Source	UniteStatesCodes(USC), Westlaw, Policies,Legal Cases, Textbooks
Data Retrieval	API Extraction and Manual
Authorization	API.ata.gov key received and authorized
Data Handling	Responses processed appropriately for storage and subsequent analysis

Key Variables - A summary of the chosen input variables (Y's) and/or output variables (X's)

	1	2	3	4	5	6
Variable title	StatesCodes(USC)/We	Policies	Textbooks	Legal Cases	Legal Proceedings	
Input (X) or output (Y) variable?	X	X	X	X	X	
Unit of measurement	N/A	N/A	N/A	N/A	N/A	
Data type	Attribute	Attribute	Attribute	Attribute	Attribute	
Collection method	Automated	Automated	Manual	Manual	Manual	
Historical data exist?	Yes	Yes	Yes	Yes	Yes	
Operational definition exist?	Yes	Yes	Yes	Yes	Yes	
Data collector	Namratha	Sohail	Yuti	Swetha	Namratha	
Start date	2-Feb	11-Apr	1-Jul	15-Sep	2-Oct	
Due date						
Duration (in days)	8	10	1	15	20	

Note. Actions made for data collection. Original from Team 1.

The datasets consist of four components: US Codes, Company Policies, Legal Cases, and Textbooks. The sizes of the datasets are as follows: US Codes are 624 MB, Company Policies are 553 MB, and Legal Cases are 645 MB. Additionally, the textbooks contribute a combined size of 220 MB. Together, the datasets amount to approximately 2 GB, providing a diverse and robust foundation for legal and corporate analysis.

Table 9

Raw Dataset Records Count for Each PDF

Chapter	Average PDF Size(Characters)	Special Characters(%)
US Code	249,314,582	5.3%
Legal Cases	260,582,349	5.89%
Policies	76,928,491	5.34%
Textbooks	26,692,695	7.23%

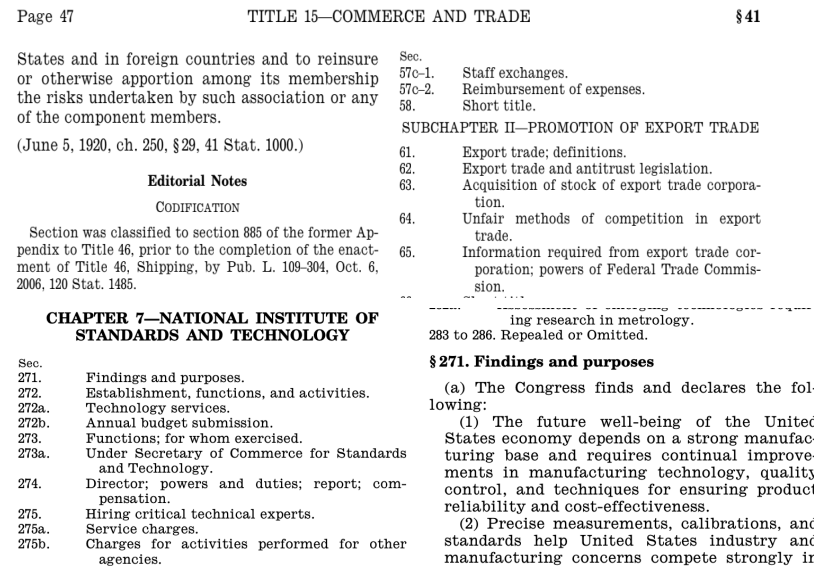
Note. Character count, and special character percentages. Original from Team 1.

Dataset Samples

Figures 12 illustrate the sample datasets obtained through API extraction. Figure 13 shows the screenshot of the complaints filed their verdicts, and the policies of companies like WhatsApp and Apple.

Figure 12

Sample Datasets



Page 47 TITLE 15—COMMERCE AND TRADE § 41

States and in foreign countries and to reinsure or otherwise apportion among its membership the risks undertaken by such association or any of the component members.

(June 5, 1920, ch. 250, § 29, 41 Stat. 1000.)

Editorial Notes

CODIFICATION

Section was classified to section 885 of the former Appendix to Title 46, prior to the completion of the enactment of Title 46, Shipping, by Pub. L. 109–304, Oct. 6, 2006, 120 Stat. 1485.

CHAPTER 7—NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

Sec.
271. Findings and purposes.
272. Establishment, functions, and activities.
272a. Technology services.
272b. Annual budget submission.
273. Functions; for whom exercised.
273a. Under Secretary of Commerce for Standards and Technology.
274. Director; powers and duties; report; compensation.
275. Hiring critical technical experts.
275a. Service charges.
275b. Charges for activities performed for other agencies.

Sec.
57c–1. Staff exchanges.
57c–2. Reimbursement of expenses.
58. Short title.

SUBCHAPTER II—PROMOTION OF EXPORT TRADE

61. Export trade; definitions.
62. Export trade and antitrust legislation.
63. Acquisition of stock of export trade corporation.
64. Unfair methods of competition in export trade.
65. Information required from export trade corporation; powers of Federal Trade Commission.

ing research in metrology.
283 to 286. Repealed or Omitted.

§ 271. Findings and purposes

(a) The Congress finds and declares the following:

(1) The future well-being of the United States economy depends on a strong manufacturing base and requires continual improvements in manufacturing technology, quality control, and techniques for ensuring product reliability and cost-effectiveness.

(2) Precise measurements, calibrations, and standards help United States industry and manufacturing concerns compete strongly in

Note. Sample extracted from Chapter 2 and Chapter 7. Original from Team 1.

Figure 13

Sample Datasets

UNITED STATES DISTRICT COURT
WESTERN DISTRICT OF WASHINGTON
AT SEATTLE

DUSTIN CURTIS, URSULA N. RILEY, HOWARD CHEN, and JAROD THRUSH, individually and on behalf of all others similarly situated,

Plaintiff,

v.

CAPITAL ONE FINANCIAL CORPORATION

and

AMAZON WEB SERVICES, INC.,

Defendants.

No. 19-cv-01366

CLASS ACTION COMPLAINT

JURY TRIAL DEMANDED

SETTLEMENT AGREEMENT AND RELEASE

This Settlement Agreement and Release ("Settlement Agreement" or "Agreement") is entered into by, between and among Alan Hall, Jeffrie Alan Summers II, and Kristina Wright, individually and on behalf of Participating Settlement Class Members (as defined in Paragraph 32) (together, "Plaintiffs"), and Defendant Sea Mar Community Health Centers (hereinafter, "Sea Mar" or "Defendant" and collectively with Plaintiffs, the "Parties"). As detailed below, this Settlement Agreement releases and forever discharges and bars all claims asserted (or that could have been asserted) in the class action lawsuit captioned *Hall v. Sea Mar Community Health Centers*, No. 21-2-15130-9 SEA ("Hall"), currently pending in the United States District Court, Western District of Washington. (the "Action") or any related actions, including, but not limited to: (i) Barnes, et al. v. Sea Mar Community Health Centers, No. 2:22-cv-00181-TLF, (ii) Lopez v. Sea Mar Community Health Centers, No. 2:22-cv-00185-MJP, (iii) Waliany v. Sea Mar Community Health Centers, No. 2:22-cv-00182-TSZ, (iv) Summers v. Sea Mar Community Health Centers, No. 2:22-cv-00183-BJR, (v) Maynor v. Sea Mar Community Health Center, No. 2:22-cv-00187-RSM (collectively, the "Related Actions").

Effective Date: January 4, 2021 (archived versions)

WhatsApp Terms of Service

Table of Contents

- About Our Services
- Privacy Policy And User Data
- Acceptable Use Of Our Services
- Third-Party Services
- Licenses

4/28/24, 11:20 AM

Legal

Legal - Global Trade Compliance - Apple

Hardware Software Sales & Support Internet Services Intellectual Property More Resources

Apple Trade Compliance

APPLE POLICY

As a global technology company, Apple is committed to complying with all applicable trade regulations in all countries in which we operate, including, but not limited to, all export and sanctions regulations. It is our policy to continually adhere to these regulations in all activities that we engage in.

Note. Sample extracted from numerous websites. Original from Team 1.

A total percentage of 5.80%, meaning 35,588,923 of the 613,518,117 characters found in the datasets, are special characters as shown in Figure 14. The complexity of the text and variability are shown by the sentence length statistics, which show an average of 13.65 words per sentence, a median of 4 words, and a maximum of 9,279 words. Because sophisticated legal and technical terms are used, the average word length is roughly 5.01 characters, with a median of 4 characters and a maximum of 102 characters. Commas, periods, and parentheses both closing and opening are the most frequently used special characters. Hyphens, colons, single quote marks, and section symbols (§) are further often used symbols as illustrated in Figure 15. The above patterns illustrate the structure of the legal, corporate, and documents.

Figure 14

Raw Data Count

```
Summary of Special Character Analysis:
  Total Characters Across All PDFs: 613518117
  Total Special Characters Across All PDFs: 35588923
  Overall Percentage of Special Characters: 5.80%
```

Note. Number of characters. Original from Team 1.

Figure 15

Statistics and Special Character

```
Sentence Length Statistics: {'Mean': 13.645318236437046, 'Median': 4, 'Max': 9279, 'Min': 1}
Word Length Statistics: {'Mean': 5.009797553629278, 'Median': 4, 'Max': 102, 'Min': 1}

Most Common Special Characters:
,: 8344057
.: 7481194
): 3815570
(: 3812925
–: 2787983
:: 2210465
‘: 1374482
’: 1163978
—: 986146
§: 851951
```

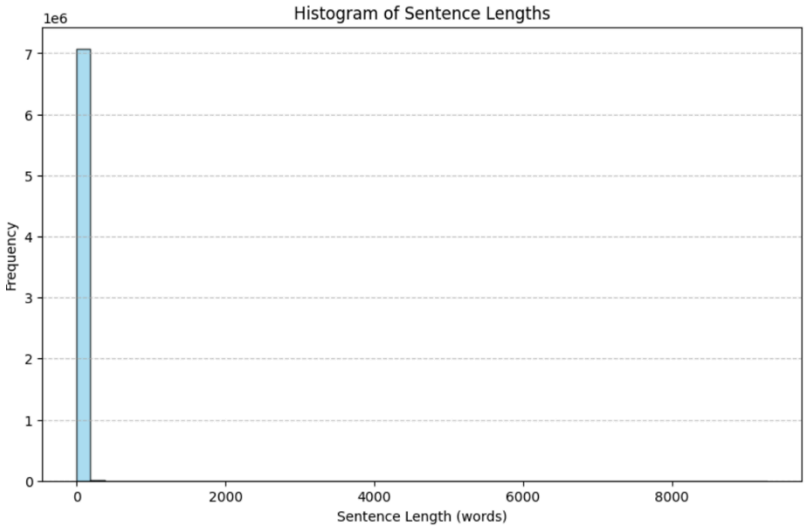
Note. Stats of word/sentence lengths with special character occurrences. Original from Team 1.

Data Pre-processing

Data pre-processes are also applied to all the PDFs. Figure 16 illustrates the distribution of sentence lengths among the examined datasets. With an apparent spike in shorter sentence lengths, the histogram shows that most sentences are short. Variability in sentence structure within the datasets is given by the distribution, reflecting the diverse nature of the documents analyzed.

Figure 16

Histogram of Sentence Length



Note. Distribution of sentence lengths. Original from Team 1.

The words that are used most frequently across the datasets are displayed in Figure 17 which depicts the word cloud. Commonly used words such as "the," "and," "of," and "section" highlight the formal and legal character of the papers. The significance of the fundamental keywords in the analyzed corpus can be seen in abundance.

Figure 17

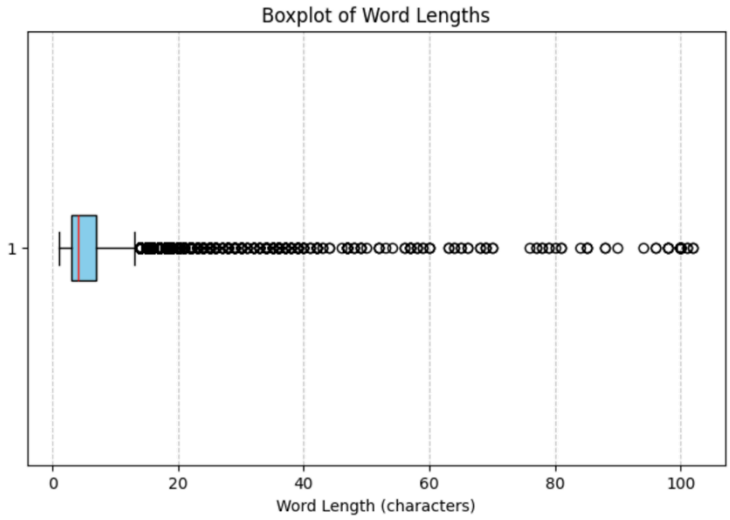
Word Cloud

Note. Common words in the corpus.Original from Team 1.

A boxplot to illustrate the variation of word lengths across the datasets is shown in Figure 18. The usage of advanced or complicated vocabulary, which is common in legal and publications, is made noticeable by the existence of outliers with lengths exceeding 50 characters, even though most phrases are between 4 and 6 characters.

Figure 18

Boxplot

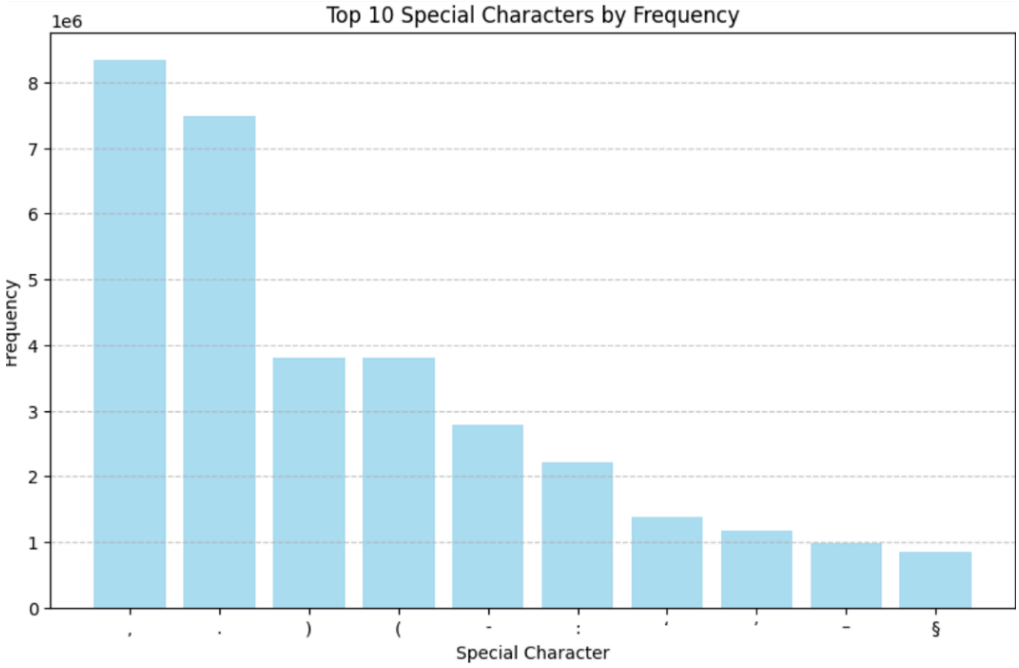


Note. Box plot for word lengths. Original from Team 1.

The ten special characters that occur most commonly in the datasets are shown in Figure 19. With almost 8 million and 7.4 million times, respectively, commas and periods predominate, followed by brackets and hyphens. The data visual highlights the crucial nature of punctuation and formatting elements that are frequently seen in corporate and legal documents.

Figure 19

Special Characters

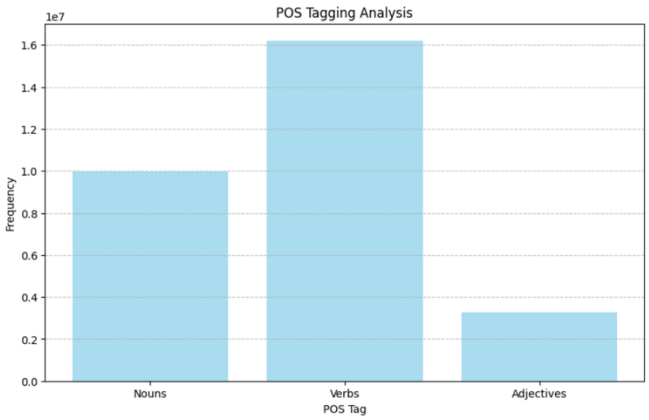


Note. The top 10 special characters count. Original from Team 1.

Figure 20 depicts how part-of-speech (POS) tags are spread throughout the analyzed text. Verbs occur the most extensively, with almost 16 million times, followed by nouns and adjectives. This distribution reflects the texts' legally binding and descriptive nature, which highlights the prevalence of action in legal and regulatory contexts.

Figure 20

POS Tagging Analysis



Note. Trends of POS tags. Original from Team 1.

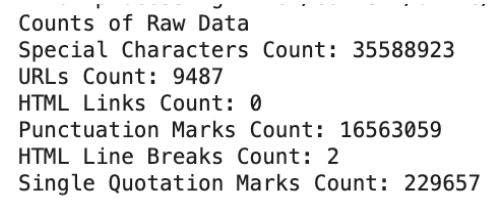
Handling Text Preprocessing

This step eliminates undesired features such as URLs, HTML links, punctuation marks, and HTML line breaks. However, crucial elements like punctuation and single quotation marks were kept to ensure the legal text's integrity. The preprocessing function, which used regular expressions (re), successfully cleansed the text data and removed redundant components from the text using substitution and pattern-matching techniques, resulting in a simplified and standardized dataset. This method ensured that the language was free of distortions and inconsistencies, laying a solid foundation for further analysis or processing.

Figures 21 and 22 are a comparison of raw and preprocessed data revealing substantial improvements in the quality of the data. The raw data contains 35,588,923 special characters, 16,563,059 punctuation marks, 9,487 URLs, and some HTML elements. After preprocessing, the total count of special characters was reduced to 28,059,812, and all URLs and HTML elements were deleted, resulting in a cleaner dataset. This helps with a more concentrated and suitable dataset.

Figure 21

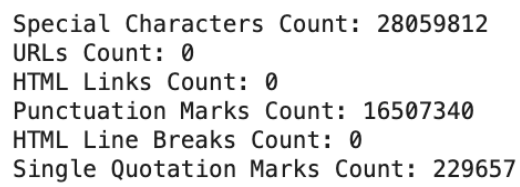
Counts of Raw Data



Note. Results of raw data count. Original from Team 1.

Figure 22

Counts of Cleaned Data



Note. Results of cleaned data count. Original from Team 1.

Lowercasing

The approach starts with importing the required libraries, which include the NLTK library for natural language processing tasks. The text data was tokenized into individual words and converted to lowercase using Python's lower() method. This provides word representation consistency and prevents redundant tokens due to case differences.

Stopword Removal

The NLTK library is used to access a large array of stopwords and tokenization tools. Stop words are extracted from the NLTK corpus. Stopwords are common words with little semantic value that can be deleted to make room for informative words. Each token in the text input is compared to the list of stopwords, and stopwords are removed to yield a list of effective words for analysis.

Pronoun Removal

A collection of pronouns, including first-person, second-person, and third-person pronouns, is established. The collection also includes possessive pronouns. Pronouns are eliminated from the list of filtered words so that the focus is on keywords with significant meaning for analysis.

Figure 23 shows the raw text retrieved from a document about commerce and trade regulations. The raw text includes section headings, numerical references, and special characters such as '§'. The raw text lacks uniformity and organization, making it difficult to undertake significant analysis or information extraction.

Figure 23

Raw Data

```
Text before preprocessing:
Page 626 TITLE 15—COMMERCE AND TRADE § 157
§ 157. Regulations and fees; disposition of fees
and penalties
(a) The Secretary is authorized to make such
regulations as may be necessary to carry into
effect the functions vested in him or in the reg -
istrar by this chapter.
```

Note. Sample of raw data. Original from Team 1.

Figure 24 displays the same text following a series of preparation procedures as discussed. Each word and numerical reference is tokenized and changed to lowercase to ensure uniformity in representation. Special characters like '§' have been eliminated to simplify the text and reduce noise. Also, words such as "registrar" have been standardized to a single form, improving the text's consistency.

Figure 24

Preprocessed Data

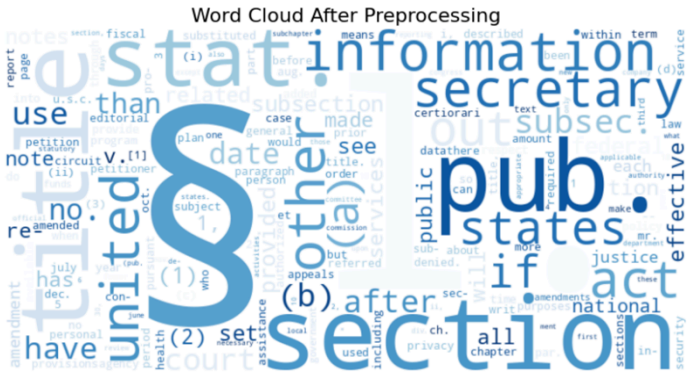
```
Text after preprocessing:
['page', '626', 'title', '15commerce', 'trade', '157', '157', 'regulations', 'fees', 'disposition', 'fees', 'penalties', 'secretary',
```

Note. Sample of preprocessed data. Original from Team 1.

Following preprocessing, the word cloud illustrates the terms that appear most frequently in the preprocessed dataset. Words like "section," "states," "information," and "secretary" point out the legalistic and ordered nature of legal and corporate documents. Symbols like "§" highlight legal references and statutory provisions, which are important to the dataset's context. Figure 25 shows how preprocessing retains important phrases and symbols while removing noise, culminating in a simpler and more precise dataset for analysis.

Figure 25

Word Count of content in Chapter 2



Note. Count of words for larger PDFs. Original from Team 1.

Data Transformation

In the data transformation phase, the cleaned data from the data cleaning phase transforms into a format more suitable to be passed as input for Large Language Models. This phase is crucial in the Data Engineering plan enabling further Natural Language processing. In this research various data transformation techniques such as Data Regularization,

Lemmatization, Question Answer pair generation, Tokenization, and Vectorization. Implementing these transformations helps improve the general quality of the data better, consistently and ultimately improves the overall performance of the LLMs.

Data Regularization

Data regularization is a technique that increases a model's accuracy with unseen data by introducing constraints or altering the training set. This can be accomplished by enhancing the current language models' training examples through the use of data augmentation approaches. Adding augmented new data to the test and validate sets will produce excessively positive evaluation results, augmentation is exclusively performed on the training dataset. To ensure that the test and validation sets contain unobserved data that the Language model analyzes, augmentation is thus implemented on the training set. As such, the preprocessed data is divided into train, test, and validation sets, with the augmentation of only the training set.

Three distinct augmentation techniques are being implemented as a part of data transformation. Augment synonym, Augment random deletion, and Augment random swap are three techniques applied to different sample sets of the training data. These tasks are being implemented using the nlpaug library to augment the training data and sentence structures, improving the input data to the Language model's ability to more generalized words in the input sentence structures.

Synonym replacement expands the dataset for question answering in Natural Language processing by replacing words in a sentence with their synonyms with new sentence structures with the same meaning. By this approach, the number of training samples for the language models is expanded by using a wide range of vocabulary, improving its ability to answer generalized user queries. To achieve this, 10% of the training data is augmented to increase the

diversity in the training samples. Sample original text data of the training dataset is shown below and Figure 26 shows the same text data with synonym augmentation has been applied.

Figure 26

Original Text from the Dataset

```
Original Text:
not more
than three of the commissioners shall be mem -
bers of the same political party. the first com -
missioners appointed shall continue in office for
terms of three, four, five, six, and seven years,
respectively, from september 26, 1914, the term
of each to be designated by the president, but
their successors shall be appointed for terms of
seven years, except that any person chosen to
fill a vacancy shall be appointed only for the un -
expired term of the commissioner whom he
shall succeed: provided, however , that upon the
expiration of his term of office a commissioner
shall continue to serve until his successor shall
have been appointed and shall have qualified..1
the president shall choose a chairman from the
commission’s membership
```

Note. Original text extracted from the dataset. Original from Team 1.

After augmenting the text, Figure 27 shows below different words of the same original text data.

Figure 27

Augmented text after Synonym-Replaced Text

```
Synonym-Replaced Text:
n three of the commissioners shall be mem - bers of the same political party. th
e first com - missioners appointed shall continue in office for terms of three,
four, five, six, and seven years,  respectively, from september 26, 1914, the te
rm  of each to be designated by the president, but  their successors shall be ap
pointed for terms of  seven years, except that any person chosen to  fill a vaca
ncy shall be appointed only for the un - expired term of the commissioner whom h
e  shall succeed: provided, however , that upon the expiration of his term of of
fice a commissioner shall continue to serve until his successor shall have been
appointed and shall have qualified..1 the president shall choose a chairman from
 the commission’s membership
```

Note. Synonym-replaced text on the original text. Original from Team 1.

Random Deletion is another useful method for data augmentation, as it randomly removes the words from the text data to increase the dataset diversity and improve data

generalization. This technique generates sentences while preserving the general context of the original text making it specifically beneficial for minimal datasets. The implementation of the random deletion can benefit the Language model to perform well on new, unseen data by providing a varied training dataset with diverse sentence structures and meanings. Figure 28 below shows the article before augmentation.

Figure 28

Original Text from the Dataset

```
Original Text:
not more
than three of the commissioners shall be mem -
bers of the same political party. the first com -
missioners appointed shall continue in office for
terms of three, four, five, six, and seven years,
respectively, from september 26, 1914, the term
of each to be designated by the president, but
their successors shall be appointed for terms of
seven years, except that any person chosen to
fill a vacancy shall be appointed only for the un -
expired term of the commissioner whom he
shall succeed: provided, however , that upon the
expiration of his term of office a commissioner
shall continue to serve until his successor shall
have been appointed and shall have qualified..1
the president shall choose a chairman from the
commission’s membership
```

Note. Original text extracted from the dataset. Original from Team 1.

Figure 29 shows after augmenting the text data with the random deletion from the original training dataset.

Figure 29

Augmented Text after Random Deletion

```
Word-Deleted Text:
not than three of the commissioners shall be mem - bers of the same political.
the first com - missioners appointed continue in for terms of three, , five,
six, and seven years, respectively, from september 26, 1914, the of each to be
designated by the president, but their successors shall be appointed for terms
of seven years, except that any person chosen to fill a shall be appointed only
for the un - expired term of the commissioner he shall succeed: provided,
however, that upon the expiration of his term of office a commissioner shall
continue to serve until his successor shall have been appointed and have
qualified. . 1 the president shall a chairman from the commission ’ s membership
```

Note. Augmented text after random data deletion.Original from Team 1.

In the third technique, the words are swapped in the training sentence to create a new sentence which again contributes to better generalization. This technique allows the model to handle the unseen data sample during the validation phase. Another 20% of the training data has been augmented with swapping words to widen the range of the words in the training dataset. Figure 30 below shows the original text as well as the augmented text after word swapping.

Figure 30

Original Text from the Dataset

```
Original Text:
not more
than three of the commissioners shall be mem -
bers of the same political party. the first com -
missioners appointed shall continue in office for
terms of three, four, five, six, and seven years,
respectively, from september 26, 1914, the term
of each to be designated by the president, but
their successors shall be appointed for terms of
seven years, except that any person chosen to
fill a vacancy shall be appointed only for the un -
expired term of the commissioner whom he
shall succeed: provided, however , that upon the
expiration of his term of office a commissioner
shall continue to serve until his successor shall
have been appointed and shall have qualified..1
the president shall choose a chairman from the
commission's membership
```

Note. Original text extracted from the dataset.Original from Team 1.

Figure 31 depicts the augmented text after random swap, the text has few swapped and jumbled sentences compared to the original text. This contributes to the numerous distinct samples of data allowing the language model to a wide range of data variations with unseen text data.

Figure 31

Augmented Text after random word-swapping

```
Word-Swapped Text:
not more than three of the commissioners shall be mem - bers of the same
political party. the first com - missioners appointed shall continue in for
office terms three of, four, five, six, and seven years, respectively, from
september 26, 1914, the term each of be to designated by the president, but
their successors shall appointed be terms for of seven years, except any that
person chosen to fill a vacancy shall be appointed only for the un - expired
term of the whom commissioner he shall succeed: provided, however, that upon the
expiration of his term of office a commissioner shall continue to serve until
his successor have shall been appointed and shall have qualified. . 1 the shall
president choose a chairman from the commission ’ s membership
```

Note. Augmented text after random data swapping. Original from Team 1.

Lemmatization

The lemmatization technique is applied to reduce words to their root form to normalize the text data. This approach uses vocabulary and morphological analysis in grouping the words of different grammatical forms to the base form so the language model can handle them as a single item. This linguistics process preserves the word meaning further helping in improving the accuracy of question-answering tasks. Figure 32 shows the original text as well as the augmented text after lemmatization.

Figure 32

Original Text from the Dataset

```
Original Text:
not more
than three of the commissioners shall be mem -
bers of the same political party. the first com -
missioners appointed shall continue in office for
terms of three, four, five, six, and seven years,
respectively, from september 26, 1914, the term
of each to be designated by the president, but
their successors shall be appointed for terms of
seven years, except that any person chosen to
fill a vacancy shall be appointed only for the un -
expired term of the commissioner whom he
shall succeed: provided, however , that upon the
expiration of his term of office a commissioner
shall continue to serve until his successor shall
have been appointed and shall have qualified..1
the president shall choose a chairman from the
commission’s membership
```

Note. Original text extracted from the dataset. Original from Team 1.

After performing the Lemmatization, Figure 33 shows the words normalized to the accurate root forms.

Figure 33

Augmented Text After Lemmatization

```
Lemmatized sentence: not more than three of the commissioner shall be mem - bers of the same
political party . the first com - missioner appointed shall continue in office
for term of three , four , five , six , and seven year , respectively , from
september 26 , 1914 , the term of each to be designated by the president , but
their successor shall be appointed for term of seven year , except that any
person chosen to fill a vacancy shall be appointed only for the un - expired
term of the commissioner whom he shall succeed : provided , however , that upon
the expiration of his term of office a commissioner shall continue to serve
until his successor shall have been appointed and shall have qualified .. 1 the
president shall choose a chairman from the commission ’ s membership
```

Note. Augmented text received after Lemmatization. Original from Team 1.

Tokenization

The texts extracted from PDFs are tokenized using the tokenizer called word_tokenize from the Natural Language Toolkit (nltk) library. This function takes a string of text as input and returns a list of individual tokens in the text. Figure 34 shows how data looks after tokenization.

Figure 34

Data after Tokenization

```
Tokens:
['page', '47', 'title', '15—commerce', 'and', 'trade', '§', '41', 'states', 'and', 'in', 'foreign', 'countries', 'and', 'to', 'reinsure', 'or', 'otherwise',
```

Note. The tokens generated after applying the tokenization. Original from Team 1.

Vectorization

Vectorization is an essential step in the data transformation process as text data are converted into a numerical format that large language models can understand better. The embeddings are generated using GloVe embedding [average_word_embeddings_glove.6B.300d] from the ‘Sentence Transformer’ library. The GloVe model trained on 6 billion tokens generates

300-dimensional vectors. Below Figure 35 shows the embedding generated using GloVe embeddings.

Figure 35

Glove Embedding Result

```
Embeddings
[[-2.64665008e-01 -2.26674005e-01 -3.47996980e-01 -2.62654990e-01
   1.97190508e-01 -9.14949998e-02  2.47254997e-01 -4.24465001e-01
  -2.78364986e-01 -1.52365005e+00  1.21467993e-01  1.36302505e-02
  -3.94914985e-01  1.76838502e-01 -3.47050056e-02 -2.98945010e-01
  -1.14455998e-01 -2.27650031e-02 -1.51705846e-01 -4.99460012e-01
   1.65081993e-01 -1.47114992e-01  1.13208249e-01  2.10548490e-01
  -1.52712509e-01 -3.78019989e-01  4.16520014e-02 -3.06959987e-01
  -1.23482503e-01 -2.67659992e-01 -3.69657487e-01  5.36225021e-01
  -4.02254999e-01  1.31069988e-01 -7.51799941e-02 -2.85070002e-01
  -1.03549957e-02 -4.88460004e-01 -3.78574997e-01  2.42608503e-01
   4.61874485e-01  3.69210020e-02 -3.33090007e-01  1.46139994e-01
  -1.51894510e-01 -1.09146953e-01 -4.75178987e-01  5.64700007e-01
   3.96899953e-02 -6.06149994e-02 -2.58730024e-01 -1.20144993e-01
   2.95935005e-01  2.06270009e-01  1.00895002e-01 -1.57559980e-02
  -1.20039999e-01 -3.19000006e-01  1.16341501e-01 -1.28049999e-02
   5.83079994e-01  1.31840006e-01  3.21864992e-01 -4.45869982e-01
```

Note. Embeddings generated using GloVe embeddings. Original from Team 1.

The generated vectors are stored in the AstraDB vector database by establishing the connection to the AstraDB database using the Astra database application token (ASTRA_DB_APPLICATION_TOKEN), Astra Database ID (ASTRA_DB_ID) and Open AI Key (OPENAI_API_KEY). Once the connection is established, the GloVe embedding is stored in the law Chapter 2 table. Figure 36 shows the output after successfully creating the embeddings.

Figure 36

AstraDB Table Creation

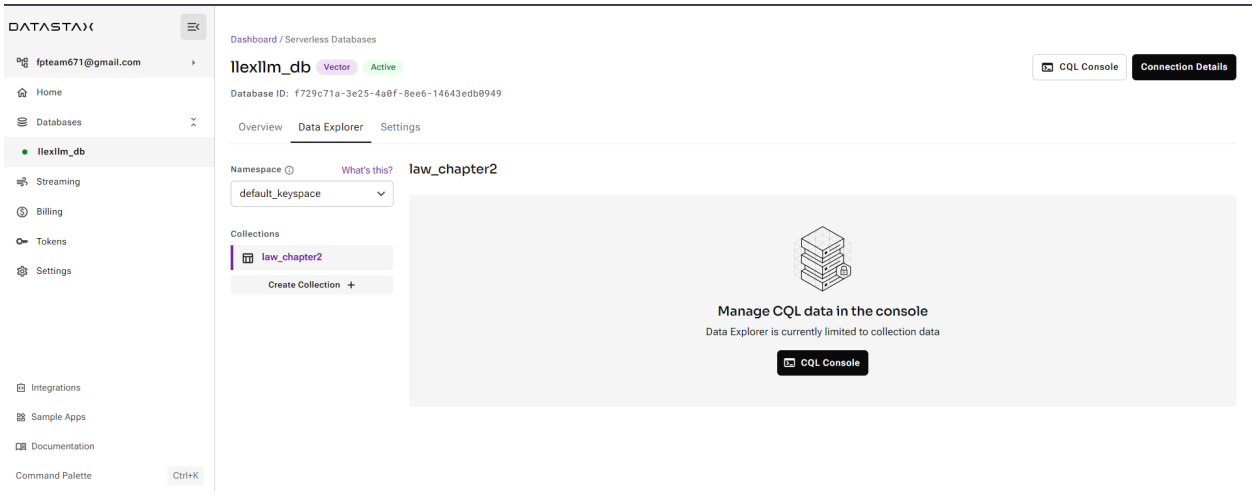
```
Table Chapter 2 created succeffully
```

Note. Creation of AstraDB table. Original from Team 1.

Figure 37 shows the Law Chapter 2 table created in the default keyspace in the AstraDB server. Figure 38 shows the outputs of the answer retrieval based on question prompts to AstraDB.

Figure 37

Astra DB Table Creation on the Server



Note. Law_chapter2 table created in the collections section under default keyspace. Original from Team 1.

Figure 38

Question Answer Retrieval

```
QUESTION: "What are the key powers and responsibilities of the Federal Trade Commission as outlined in the search results?"
WARNING:cassandra.protocol:Server warning: Top-K queries can only be run with consistency level ONE / LOCAL_ONE / NODE_LOCAL. Consistency level LOCAL_QUORUM was requested. Downgrading the consistency level to LOCAL_ONE.
ANSWER: "Some of the key powers and responsibilities of the Federal Trade Commission include enforcing section 18(a)(1)(B) of the Federal Trade Commission Act, utilizing the same powers and jurisdiction as the Federal Trade Commission Act, and imposing penalt

FIRST DOCUMENTS BY RELEVANCE:
WARNING:cassandra.protocol:Server warning: Top-K queries can only be run with consistency level ONE / LOCAL_ONE / NODE_LOCAL. Consistency level LOCAL_QUORUM was requested. Downgrading the consistency level to LOCAL_ONE.
    [0.9236] "under section 18(a)(1)(B) of the Federal Trade
Commission Act (15 U.S.C. 57a(a)(1)( ..."
    [0.9236] "under section 18(a)(1)(B) of the Federal Trade
Commission Act (15 U.S.C. 57a(a)(1)( ..."
    [0.9230] "means, and with the same jurisdiction, pow -
ers, and duties as though all applicabl ..."
    [0.9230] "means, and with the same jurisdiction, pow -
ers, and duties as though all applicabl ..."

What's your next question (or type 'quit' to exit): What is the purpose of the Federal Trade Commission as established in the search results?

QUESTION: "What is the purpose of the Federal Trade Commission as established in the search results?"
WARNING:cassandra.protocol:Server warning: Top-K queries can only be run with consistency level ONE / LOCAL_ONE / NODE_LOCAL. Consistency level LOCAL_QUORUM was requested. Downgrading the consistency level to LOCAL_ONE.
WARNING:cassandra.protocol:Server warning: Top-K queries can only be run with consistency level ONE / LOCAL_ONE / NODE_LOCAL. Consistency level LOCAL_QUORUM was requested. Downgrading the consistency level to LOCAL_ONE.
ANSWER: "The purpose of the Federal Trade Commission, as established by the cited text, is to grant the Commission statutory authority to enforce laws and seek injunctive relief in investigations related to the petroleum industry and other matters that protec

FIRST DOCUMENTS BY RELEVANCE:
    [0.9261] "troleum industry, as well as in other major investiga -
tions designed to protect th ..."
    [0.9261] "troleum industry, as well as in other major investiga -
tions designed to protect th ..."
    [0.9215] "he may deem proper.
For the purpose of enforcing these provisions
the Federal Trad ..."
    [0.9215] "he may deem proper.
For the purpose of enforcing these provisions
the Federal Trad ..."
```

Note. Sample output of the relevant documents retrieved by AstraDB. Original from Team 1.

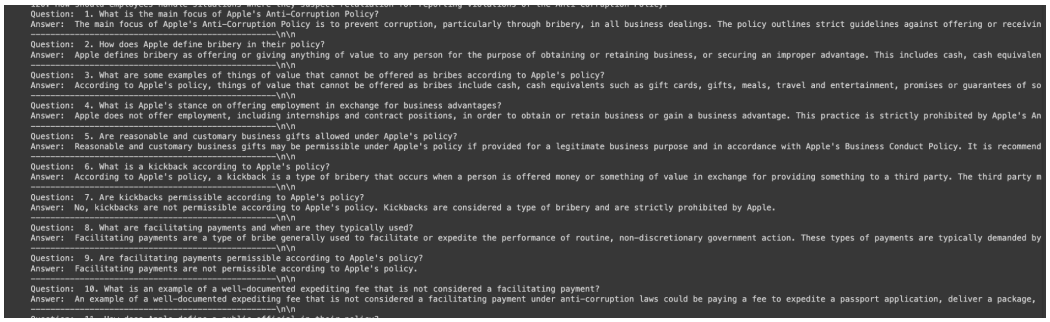
By storing the embeddings of the documents, the vector search retrieves the relevant data from the documents. The relevant documents will be passed to the language models along with the user queries to generate the user response.

Data Preparation

Careful data preparation is essential in developing the PDF chat application specialized in the legislative sector. After transforming the PDFs, the next crucial step involves preparing training, validation, and test datasets from these transformed datasets. This procedure encompasses structuring the textual data into prompt-response (question-answering) pairs to optimize the large language models with the proposed model. The questions are constructed based on the key concepts and topics derived from the document and the corresponding answers are extracted from the relevant chunk or section within the text providing contextual knowledge. This methodical approach makes it easier for the LLMs to be trained, validated, and evaluated. Considering the same instances(sample data) that have been used in the earlier processes, before splitting the data, the total question-answer pairs are 19,983. Figure 39 shows the sample question-answer pairs dataset generated to train the model.

Figure 39

Sample Question-Answer pair generated



Note. Question-answer pairs are generated from the PDF named Anti-Corruption_Policy-Apple. Original from Team 1.

Data can be distributed easily for training, testing, and validation, by using a consistent split ratio, such as 80:10:10. The LLMs can learn from a significant amount of the data where 80% of the prepared datasets are assigned to the training set. 10% of the datasets make up the validation set, which is an essential part of the model optimization process that helps with overfitting reduction. The test set receives the remaining 10%, offering an unbiased result. Figure 40 below shows the split of data for fine-tuning the LLMs.

Figure 40

Available Question-Answer pairs in each set

```
Training set length: 17241
Validation set length: 1271
Testing set length: 1271
```

Note. Sample number of available QA pairs for LLM fine-tuning. Original from Team 1.

Figure 41 represents the training data set sample data. A total of 17,241 QA pairs are available for the training set.

Figure 41

Showing sample from the Training dataset

```
=== Training Set ===
Question 1: What is mentioned in the document about L. 109–455,
m...
Answer 1: L. 109–455, see Termination Date of 2006 Amend –
ment note below.
```

Note. Displaying the sample of the training set. Original from Team 1.

Figure 42 represents the testing data set sample data. A total of 1271QA pairs are available for the testing set.

Figure 42

Showing sample from the Testing dataset

```
=== Testing Set ===
Question 1: What is mentioned in the document about CODIFICATION
Section was enacted as part of the ...
Answer 1: CODIFICATION
Section was enacted as part of the Opioid Addiction
Recovery Fraud Prevention Act of 2018, and also as part
of the Substance Use—Disorder Prevention that Pro –
motes Opioid Recovery and Treatment for Patients and
Communities Act, also known as the SUPPORT for Pa –
tients and Communities Act, and not as part of the
Federal Trade Commission Act which comprises this
subchapter.
```

Note. Displaying the sample of the testing set. Original from Team 1.

Figure 43 represents the validation data set sample data. A total of 1271 QA pairs are available for the validation set.

Figure 43

Showing sample from the Validation dataset

```
Question 2: What is mentioned in the document about Any person receiving such notification
may file w...
Answer 2: Any person receiving such notification
may file with the appropriate district court or
court of appeals of the United States, as appro –
priate, an application for a stay of disclosure.
```

Note. Displaying the sample of the validation set. Original from Team 1.

Data Statistics

Cleaning and standardizing text data taken from court cases listed in different USC chapters and other company policies PDFs was part of the data preparation procedure. At first, the unprocessed data was disorganized and inconsistent, with a variety of textual elements including punctuation, special characters, and section titles. The study used several preprocessing techniques and transformations to overcome these issues. Table 10 below highlights the statistics of the data at the end of each stage.

Table 10

Summary of Statistics

Dataset	Statistics and Summary
Raw PDF Files	The source data set is made up of 624 MB of US codes, 553 MB of company policies, 645 MB of court cases, and 220 MB of textbooks. Average word count of ~40,000 words per file. Capabilities, which make it appropriate for handling intricate legal laws
Pre-processed Text	Total corpus of about 1 Million characters and word length of about 200 thousand words.
Transformed Text	About 80% of the original data size. Most stopwords and Special characters were removed. Punctuations and section structure preserved.
Prepared Question-Answer pairs	Train set- 17241 QA pairs, Test set- 1271QA pairs, Validation set- 1271 QA pairs

Note. The table shows the summary of statistics at the end of each stage. Original from Team 1.

Box plots and histograms are two examples of statistical presentations that were used to show the results of the data preparation. Histograms were used to show the word count distribution of the preprocessed dataset, and box plots comparing sentence and word lengths were used to emphasize any differences between chapters.

The unprocessed PDF files that were collected from the US Government website have a variety of features. The overall data collected is about 2GB in size. Word counts range greatly, with an average of about 60,000 words per document and a range of 2,000 to over 300,000 words. While ordinary symbols like '*', '†', and '‡' occur less frequently, about 5 times every page, special characters like '§' are common, appearing approximately 15 times per page on average. The most often used punctuation marks are commas, periods, and parentheses. Commas are used on average 20 times every page, while periods and parentheses are used 15 times each. Chapter numbers and brief content descriptions are often indicated in document titles, which

typically indicate creation dates between 2010 and 2022. Text complexity measurements show that the text is appropriate for high school graduates and has a moderate level of linguistic complexity, with an average Flesch-Kincaid Grade Level of 12 and a Gunning Fog Index of about 14. About 90% of the content is in English, while small amounts may also be in Spanish or another language. English is the dominant language. With an average of ten section headings and twenty subsections in each document, the publications follow a hierarchical structure that facilitates content organization through paragraph breaks, subsections, and section headers.

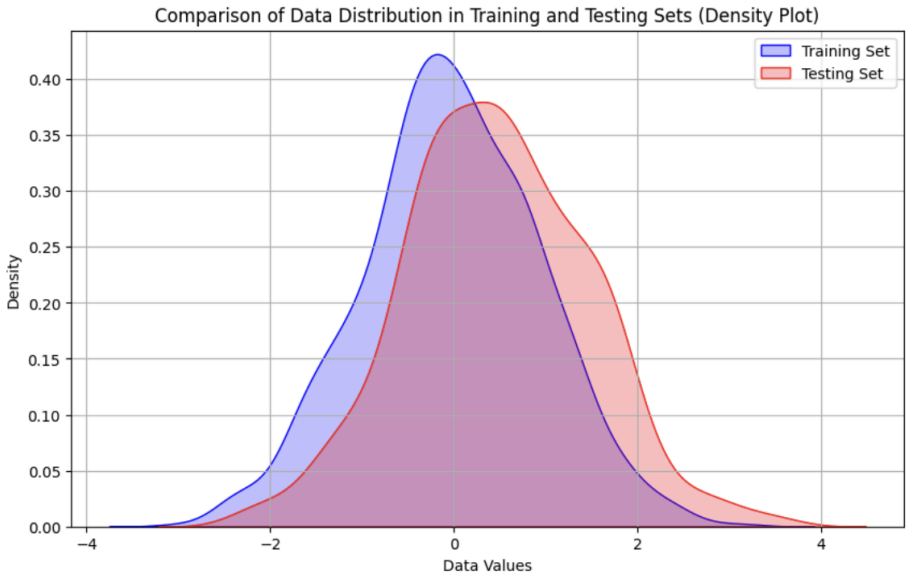
Substantial alterations are made to the raw PDF files after preprocessing and transformation. With the modified files averaging about 20% of the size of the original PDFs, the changed data—typically kept in text format—sees a significant reduction in file size. With an average of 40,000 words per document, the dataset's overall word count drops to about 80% of its initial amount. The modified data has special characters like '¨', '*', '†', and '‡' eliminated, leading to a more consistent and tidy text. Punctuation is still used, such as commas, periods, and parentheses, but some punctuation—like single quotation marks—may be removed during preprocessing, causing its usage to somewhat drop. Comparing the modified data to the raw PDF files, there may be a minor decrease in the Gunning Fog Index measurements, suggesting easier readability and less linguistic difficulty. The language distribution is still primarily English; any multilingual content will probably be kept but standardized before being processed. The converted data retains the documents' hierarchical structure, including paragraph breaks, section titles, and subsections, but with uniform formatting elements for uniformity.

The prepared question-answer pairs dataset consists of 17241 in the training set, 1271 samples in the test set, and 1271 samples in the validation set. These sets are ready for use in

training, testing, and validating machine learning models for question-answer tasks. Figure 44 below compares the data distribution of words in the training and testing sets of the QA pairs.

Figure 44

Data distribution of words in the Training and Testing sets

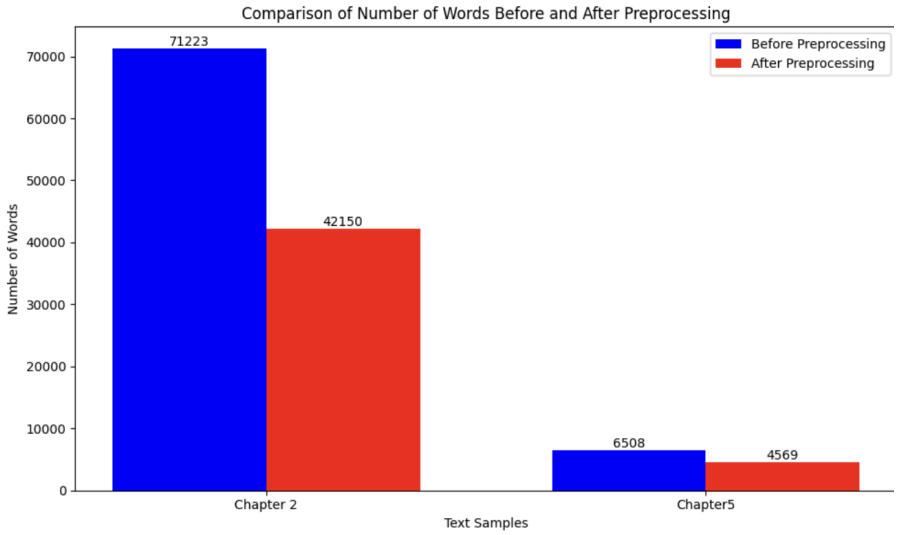


Note. Comparison of the distribution of words in training and testing sets. Original from Team 1.

Figure 45 below shows the comparison of the number of words in two example chapters before and after text processing. We see an average of 20% reduction in the number of words.

Figure 45

Word Count Split Before and After Processing.



Note. Comparison of the number of words before and after preprocessing. Original from Team 1.

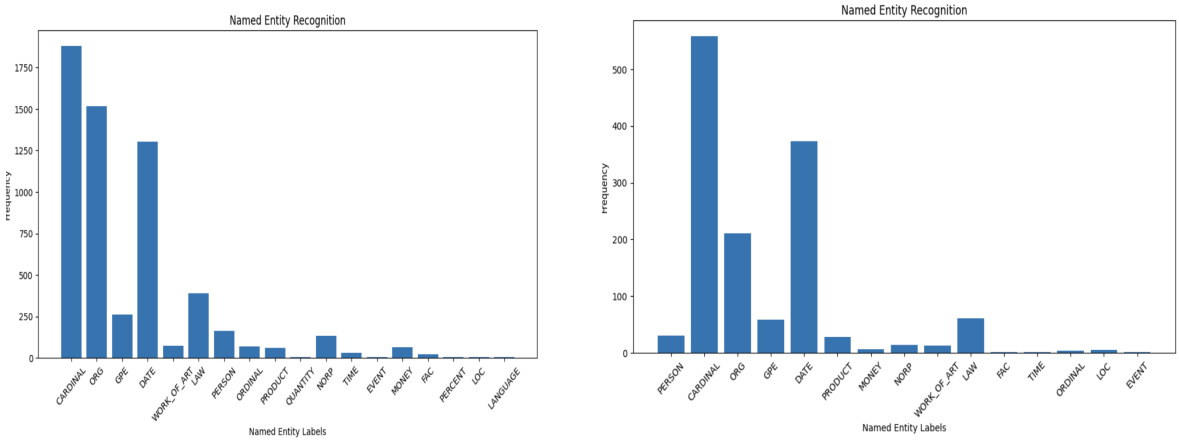
Data Analytics Results

By identifying patterns, correlations, and trends in intricate datasets, our data analytics findings offer practical insights. These results facilitate well-informed decision-making, streamline procedures, and pinpoint areas for expansion. We provide accurate and significant findings that are suited to business requirements by utilizing cutting-edge analytics methodologies.

Figure 46 below shows the results of named entity recognition for two of the chapters. The Word ‘Cardinal’ is by far the most recognized entity and can lead to bias.

Figure 46

Results of named entity recognition

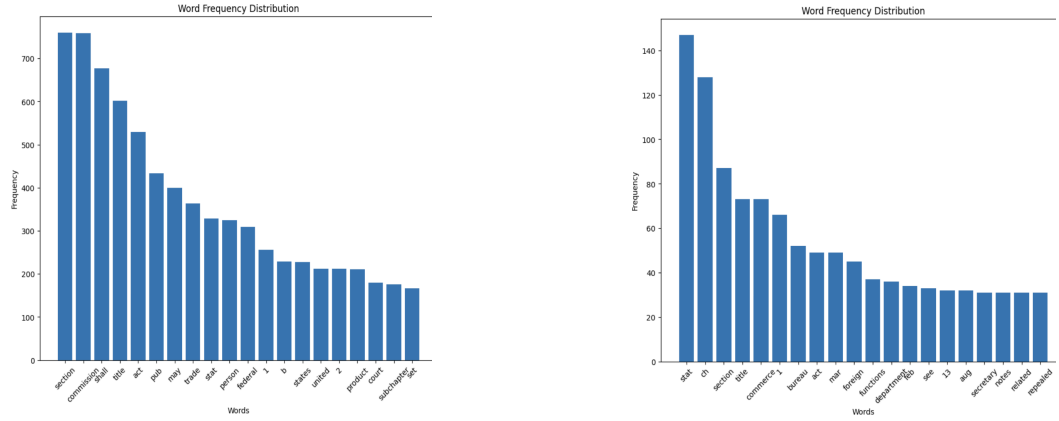


Note. The two charts above show the results of named entity recognition for two of the chapters. Original from team 1.

Figure 47 below shows the results of Word frequency distribution for two of the chapters. Figure 48 below shows the word cloud for these words. The words ‘section’, ‘commision’, and ‘stat’ are by far the most common and can lead to bias.

Figure 47

Word frequency distribution



Note. The charts above show the Word frequency distribution of two of the chapters. Original from team 1.

The words that appear most frequently in our database of legal papers are visualized by the word cloud. Words like "the," "of," "and," "to," "for," and "by" are common in legal language, which is formal and strongly dependent on conjunctions and prepositions to provide exact and thorough wording. Other often used words like "section," "subsection," "State," "Act," and "United" imply the emphasis on regulatory settings, statutory references, and the governmental or geographic boundaries that are common in legal content. The linguistic patterns present in legal documents, which are essential to their purpose and interpretation, are highlighted by this word cloud. These had to be removed for better clarity. Figure 48 shows the word cloud.

Figure 48

Word Cloud

Note. Word cloud representation for one of the chapters. Original from team 1.

Model Development

Model Proposals

Generative Pre-trained Transformer 4o (GPT -4o)

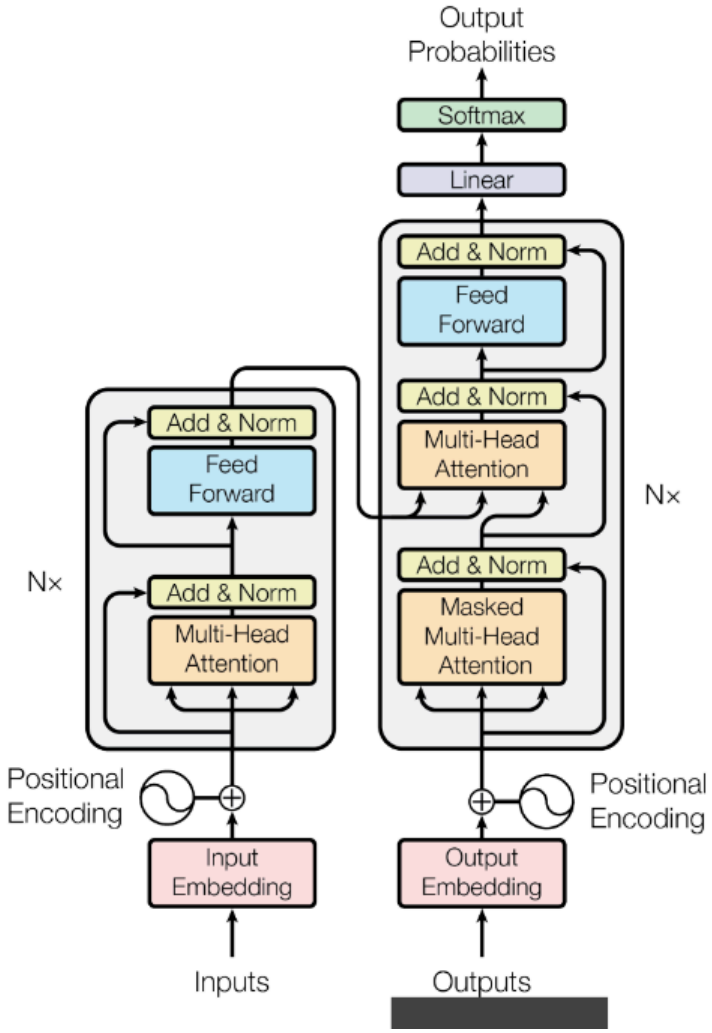
The OpenAI Team (2023) introduced the GPT-4o model. It is the language model that is autoregressive. Its main architecture is based on the concept of the Transformer which was first

introduced by Vaswani et al. (2017). It only focuses on and uses the decoder side of the Transformer. Similar to its predecessor, GPT-4o's architecture relies primarily on the mechanism of self-attention. It contains several layers with each layer consisting of two main sub-layers namely the feed-forward neural networks and the self-attention layer. In the self-attention layer, the model is provided with the capability to focus on different parts of the input sequence to get a thorough understanding of the context and to understand relationships between different words. The feed-forward neural network is used to process the output generated from the attention layer to generate the intermediary representation.

The output from the component consists of self-attention and a feed-forward neural network undergoes the process of layer normalization which is further added with the residual block to stabilize the process of learning and allowing usage of deeper layers without the issue of vanishing gradient. The final layer of the model is a linear layer that projects the generated output from the decoder to the vocabulary which is converted into the probability value of each token in that vocabulary using the softmax function. Figure 49 depicts the architecture of the transformer model which set the basis for the GPT-4o model.

Figure 49

The architecture of the Transformer model



Note. The architecture of the Transformer model From "Attention is All You Need" by Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I., 2017. https://doi.org/10.48550/arXiv.1706.03762. Copyright 2017 by Creative Common Attribution.

The self-attention mechanism which is the basis of the architecture calculates the attention score by comparing each word present in the input text with every other word. The operation is performed by creating the projection of each token present in the sequence on three vectors namely queries (Q), keys (K), and values (V). The projections are performed by

multiplying the embeddings created for the input sequence with the weights learned during training. The attention score is calculated by performing the dot product of the query vector with all the key vectors. The scores help to drive the model to determine how much attention should be given to each component of the input words. The normalization of the obtained score is performed using the softmax function to differentiate the importance of the score and output is generated for the single head as the weighted sum of the value (V) vectors which is shown in Equation 1.

$$Attention\ (Q.K,V)\ =\ softmax\ (\frac{QK^T}{\sqrt{d_k}}) \tag{1}$$

where d_k represents the dimensionality value of the key (K) vectors.

In GPT-4o multiple sets of queries, keys, and value vectors with each set having different projections that were learned, are used to allow the model to accept the information obtained from different subspaces simultaneously which helps the model to capture the relationships of different types present between words efficiently and correctly which is represented in Equation 2 and shown in Figure 49.

$$MultiHead\ (Q,K,V)\ =\ Concat(head_1,\ head_2,\ \ldots,\ head_n).W^O \tag{2}$$

where W^O represents the weights of the output matrix.

The objective of training the model is to reduce the negative log-likelihood value of the data and maximize the probability value of the next token being generated correctly. Equation 3 θ represents the total number of trainable parameters available in the model which also includes the weights and biases involved at different layers of the model, $context$ represents the sequence of tokens used to generate the next token except for the target token, $target\ token$ represents

the token that the model is trying to predict, and L(θ) represents the loss function that is needed to be minimized which is shown in Equation 3.

$$L(\theta) = -\Sigma log\ P(\frac{target\ token}{context;\ \theta}) \quad (3)$$

Large Language Model Meta AI 3.1 (LLaMA 3.1)

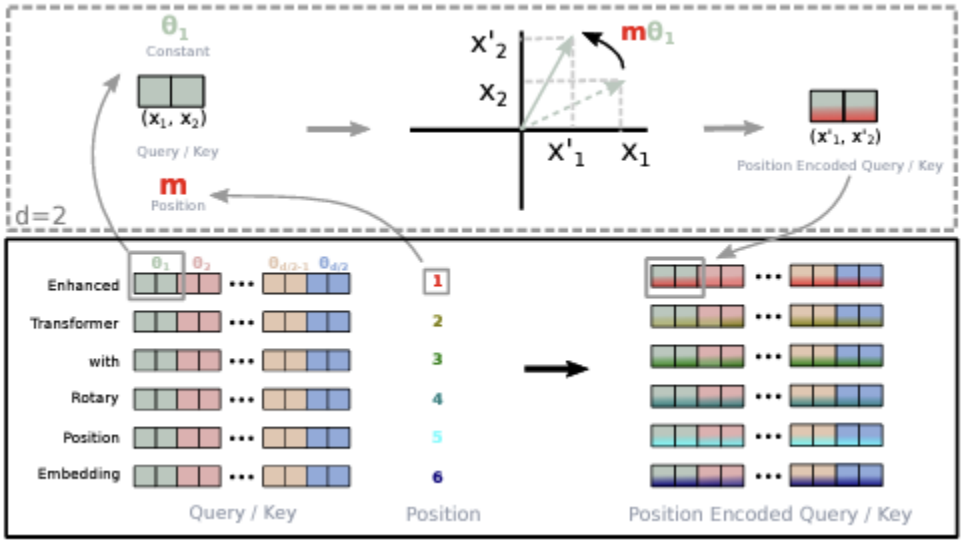
Touvron et al. (2023a) proposed LLaMA. It is also an autoregressive-based decoder-only large language model that is based on the architecture of the transformer like GPT. The model incorporates the improvements that were suggested in other pre-existing LLM models. Some of the noticeable changes in the architecture were the usage of the variant of normalization named Root Mean Square (RMS) Layer Normalization (Zhang and Sennrich, 2019) which is applied before the major layers like the self-attention layer or feed-forward neural network layer of the transformer block of the model which is represented in Equation 4, application of SwiGLU (Shazeer, 2020) activation function instead of ReLU function in feed-forward neural network layers which requires multiplication of three matrices shown in Equation 5, and utilizing Rotary Position Embedding (RoPE) (Su et al., 2021) instead of absolute or relative positional embedding during the initial embedding by encoding the absolute position of the token with the help of rotation matrix and adding the relative position of the token directly into the self-attention layer which is shown in Figure 50.

$$\overline{a}_i = \frac{a_i}{RMS} \quad \text{where} \quad \text{RMS} = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n} a_i^2} \quad (4)$$

$$\text{SwiGLU}(x) = \text{Swish}(sW).\ xV \text{ where Swish}(sW) = sW.\ \text{Sigmoid}(\beta sW) \quad (5)$$

Figure 50

Implementation of RoPE



Note. The overview of the implementation of Rotary Position Embedding. From "RoFormer: Enhanced Transformer with Rotary Position Embedding." by Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., & Liu, Y., 2021, *In the Proceedings of the 2019 Conference on Neural Information Processing System.* https://doi.org/10.48550/arXiv.1910.07467. Copyright 2019 by Creative Common Attribution.

Most of the other parts of the architecture of LLaMA 3.1 are similar to that of the GPT like input embedding, usage of a linear layer as the output layer, etc. The major difference that was observed between the initial and second/third versions of LLaMA was the increase in context length from 2K to 4K (in LLaMa 2), 8K (in LLaMa 3), and 128K (in LLama 3.1) tokens and the introduction of the concept of grouped query instead of multi-head in the attention mechanism in LLaMA 2 (Touvron et al., 2023b) and LLaMa 3 (Llama Team, 2024). In grouped-query attention (GQA), the single key (K) and value (V) vectors are used for each set or group of query (Q) vectors which is somewhere between multi-head and multi-query attention, it provides the model the capability to process multiple queries at the same time rather than processing only one.

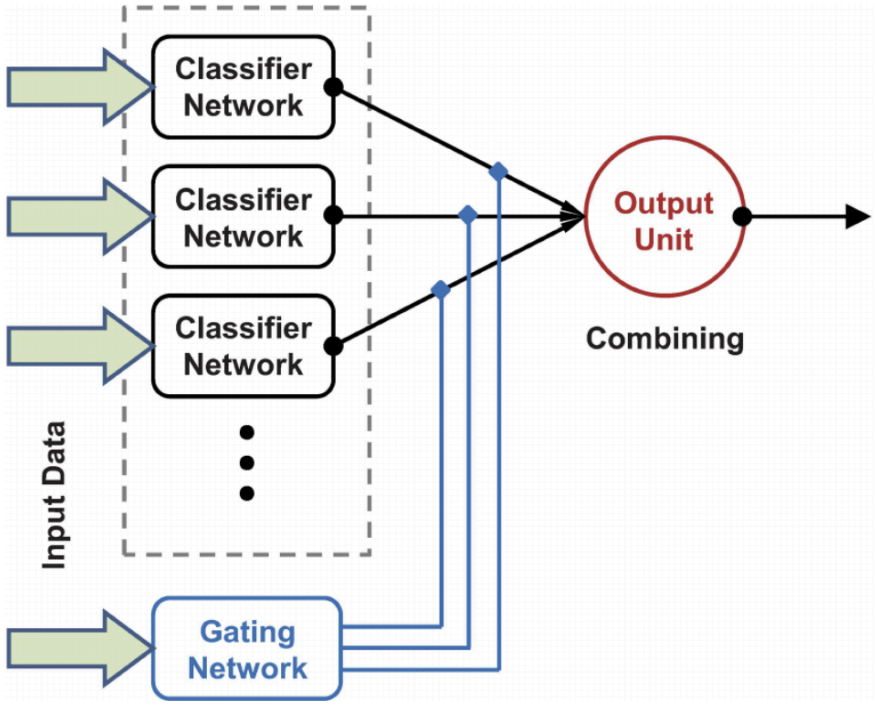
Gemini 1.5

Gemini Team (2023) introduced the language model based on the architecture of the Transformer named Gemini which is similar to other LLMs like GPT. It is launched in the Ultra, Nano, and Pro series. It also contains multiple decoder layers stacked together where each layer contains two sub-layers: a self-attention layer and a feed-forward neural network layer. Each of these layers performs the same operations as it does in the GPT model then layer normalization is also performed on it. The output from the final decoder block is provided to the linear layer paired with the softmax function to provide probabilities of the final prediction. The objective of the Gemini 1.0 model is the same as the GPT model to reduce the loss function by increasing the probability of correctly generated tokens.

Gemini Team (2024) improved the existing Gemini 1.0 model and introduced the Gemini 1.5 model that is having an architecture similar to that of Gemini in some layers but instead of being a transformer decoder model, it is a multi-modal model where the decoder layer is replaced with the mixture of expert (MoE) layer. It brings in the concept of dynamic routing that doesn’t activate all the available parameters rather it activates the subset of parameters (also known as experts) based on the input sequence provided by the user. Over the period the model learns to activate only the most relevant experts to generate contextually relevant outputs. The general architecture of the MoE layer is illustrated in Figure 51.

Figure 51

Architecture of MoE layer



Note. The general architecture of different components involved in the MoE layer. From "Unsupervised Learning in an Ensemble of Spiking Neural Networks Mediated by ITDP" by Shim, Y., Philippides, A., Staras, K., & Husbands, P., 2016, *PLOS Computational Biology, 12*(10). http://dx.doi.org/10.1371/journal.pcbi.1005137. Copyright 2016 by Creative Common Attribution.

In the MoE layer, the concept of gating plays a pivotal role in determining the relevance of each of the experts for a provided input (x). This operation is performed with the help of the softmax function where each expert is assigned a particular weight (w_i) which is shown in Equation 6.

$$w = softmax\,(G(x)) \tag{6}$$

Here, the G(x) represents a small trainable function or gating function that usually maps the input (x) to a score obtained for each expert, and softmax (G(x)) represents the softmax function

applied to each of these obtained scores to ensure that set of weights (w) will have the sum as 1 which can be seen in Equation 7.

$$softmax\,(z_i) = \frac{e^{z_i}}{\Sigma_j\, e^{z_i}} \tag{7}$$

Each of the experts present in the set (E_i) processes the input mutually exclusive of each other to produce the output (o_i) which can be seen in Equation 8.

$$o_i = E_i(x) \tag{8}$$

The final output generated in the MoE layer after processing inputs by each of the experts is the weighted sum value of the outputs from each of the experts where weights are determined by the gating mechanism in Equation 6 which can be seen in Equation 9.

$$y = \Sigma_i w_i\, o_i \tag{9}$$

Mixtral 8x7b

Jiang et al. (2024) proposed Mixtral 8x7b, a decoder-only transformer-based large language model similar in architecture to the Mixtral 7b introduced by Jiang et al. (2023). In this, each layer consists of a combination of attention mechanisms followed by the feed-forward network which is close to that of the GPT. Here, similar to most of the other LLMs the input is the sequence of tokens that are embedded in the form of vectors and parsed through each of the layers to generate the probability of position of each token. The most noticeable change concerning all the previously discussed models is the introduction of the Sparse Mixture of Expert layers (SMoE). Each layer consists of 8 blocks of independent feed-forward network known as “experts”. The experts are selected based on the router network that picks two experts for each token at each SMoE layer and after the processing is performed the output generated from each of them is combined. To decide which expert to pick, the router network utilizes the

gating mechanism that determines which experts will be most suitable for the particular task at hand and meet the characteristics of that token. The general architecture of the MoE layer is shown in Figure 3 and the SMoE layer also has a similar architecture but the selection of experts is just more sparse. The rest of the architecture is very similar to the previously discussed model like Gemini 1.5. The output generated from each of the MoE layers can be defined as a weighted sum of the outputs generated from the experts which is shown in Equation 10.

$$y = \Sigma_{i=0}^{n-1} G(x)_i . E_i(x) \quad (10)$$

Here, y is the output generated from the MoE layer, $G(x)_i$ is the i-th component of the output generated from the gating network mechanism, x is the input, and $E_i(x)$ is the output generated from the i-th expert network. Each of the gating networks G(x) calculates the score obtained by each of the experts and then top K logits (l) are selected for the particular input (x) by applying the Softmax function on it which can be seen in Equation 11.

$$G(x) = Softmax(TopK(x.W_g)) \quad (11)$$

Here, the W_g represents the weight matrix utilized by the gating network and $TopK(x.W_g)$ represents the operation performed to retain the values of Top K logits (that are used to select the experts) and set the values of all the others to negative infinity. In the case of Mixtral K=2. Each of the experts selected uses the SwiGLU activation function shown in Equation 5 to perform the operations. The final output (y) is the combination of the operations performed by SwiGLU and the gating network ($G(x)$) which can be seen in Equation 12. This is a more detailed version of the Equation 10.

$$y = \Sigma_{i=0}^{n-1} Softmax(TopK(x.W_g))_i . SwiGLU_i(x) \quad (12)$$

Model Supports

Environment, Platform, and Tools

Large Language models are known for their complex structure, and they contain a huge number of layers and parameters (in billions) that require platforms, tools, and environments that can handle the execution process of fine-tuning these models effectively. To ensure complete operations that are involved in the process like training the model, and testing it against various scenarios and evaluation metrics are performed appropriately and proper allocation of resources is ensured for each of such tasks involved, a laptop with the specifications of 14-core CPU and presence of an Intel Core i7 processor with 2560 CUDA core GPU is decided to be used. It consists of 6 performance cores and 8 efficiency cores with 12 GB RAM in CPU and 8 GB dedicated RAM in GPU. To handle fine-tuning the model on a large amount of data the GPU present in the GPU lab is decided to be used with the specifications of a 16-core GPU and 128 GB RAM. The laptop is installed with the latest version of Windows 11 Professional operating system.

To meet the requirements of the resources and to ensure that proper computation is carried out the Jupyter Notebook with the version of Python 3.7.11 and Google Collab cloud server with the hardware accelerator as T4 GPU having the RAM of 23.1 GB and the disk space of 78.2 GB are decided to be used. A small amount of PDF documents are executed on the Jupyter Notebook to check for any potential issues in the documents; to handle a huge number of PDFs and to allow the faster process of fine-tuning the Google Collab is used. It has more available resources and higher computational power which makes it ideal for complex processes. Table 11 and Table 12 illustrate the software and hardware specifications respectively.

Table 11

Software Specifications of the Project

Tools	Purpose	License
Jupyter Notebook	Web-based environment for the execution of the LangChain framework	BSD-3 Clause
Google Collab	Cloud-based environment for the execution of the LangChain framework	Proprietary
AstraDB	Vector Database	Proprietary

Note. List of software specifications required for fine-tuning models. Original from Team 1.

Table 12

Hardware Specifications of the Project

Hardware	Memory (RAM)	Configuration	Purpose
14-core CPU and 2560 CUDA Core GPU	12 GB and dedicated 8GB	6 performance core 8 efficient cores	For different LLM frameworks and performing development and fine-tuning
16 Core GPU	128 GB		For fine-tuning LLM models.
T4 GPU	16 GB		For model training, fine-tuning and testing

Note. List of hardware specifications for training and fine-tuning models. Original from Team 1.

To handle the word embeddings created from the PDF documents the cloud-based Cassandra-based vector database named Astra DB is used. It allows faster vector search and handles huge amounts of data duly which makes it suitable for complex operations like fine-tuning the LLM models.

Python is one of the most commonly used programming languages that includes a great set of libraries that can be used for performing different tasks like data cleansing, data transformation, storing the chunks into vector databases, performing RAG, and fine-tuning the LLM models. To accomplish different sets of mentioned tasks the Python version 3.7.11 is used. Initially, libraries like PyPDF2, NLTK (Natural Language Toolkit), and RE (Regular Expression) are used for initial data loading, data cleaning, and data transformations. Some of the packages and frameworks like LangChain, OpenAI, and CassIO are used to convert and store chunks of textual information into a vector database and retrieve it using RAG to generate responses to user queries. Table 13 shows the list of libraries used to perform different sets of tasks with their corresponding purposes.

Table 13

Libraries Used for Data Preparation, and Storage

	Library	**Method**	**Purpose**
PyPDF2	PyPDF2	PdfReader	To open, read, and manipulate the content in PDF files
Regex	re	sub	To replace the existing string in a particular pattern with the new replacement string
		findall	To find all the occurrences of a particular pattern in the string
NLTK	nltk.corpus	stopwords	To access the list of all the stopwords
		wordnet	To access WordNet and understand the semantic relationship between words
	nltk.tokenize	sent_tokenize	To tokenize the textual content of the PDF documents into sentences

	Library	Method	Purpose
	nltk.tokenize	word_tokenize	To tokenize the string or sentences into individual words
	nltk.tag	pos_tag	To assign part of speech to each of the words present in the given text
	nltk.stem	WordNetLemmatizer	To reduce the words into the smallest form known as the lemma
Textwrap	textwrap	fill	To wrap the textual information as a block with a specified width
LangChain	langchain.llm	OpenAI	Using the OpenAI API key to access the LLM model.
	lanchain.embeddings	OpenAIEmbeddings	To perform word embeddings of given textual information using OpenAI Embeddings
	langchain.text_splitter	CharacterTextSplitter	To split the text into chunks to maintain the token size
	langchain.vectorstores .cassandra	Cassandra	To perform vector store of the created embeddings, converting user query into vector stores by creating embedding, and performing similarity search of generated response with the document content

	Library	Method	Purpose
	langchain.indexes.vectorstore	VectorStoreIndexWrapper	To create a wrapper around the vector store for easy access
CassIO	cassio	init	To initialize the connection with the Astra vector database

Note. List of the libraries to perform data cleaning, transformation, and storage. Original from Team 1.

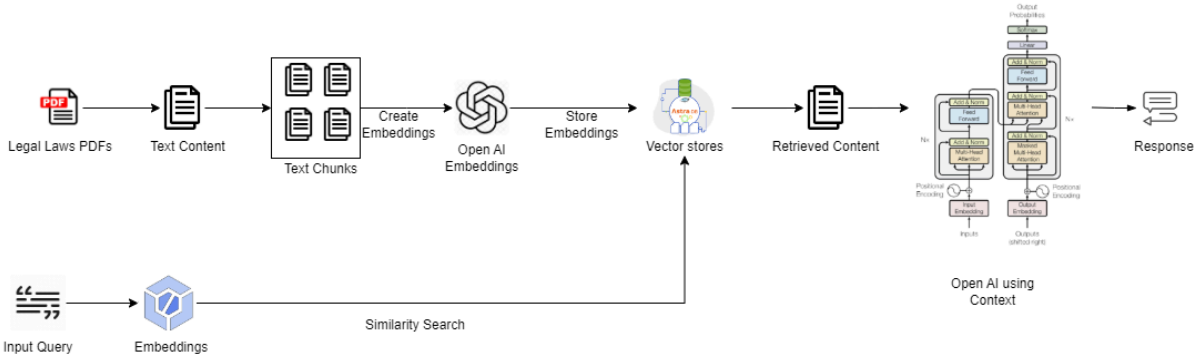
RAG-based GPT-4o Model

The data flow diagram involving Open AI's GPT-4o is illustrated in Figure 52 which shows the flow of data through different components. Initially, the legal laws pdf documents were pre-processed and converted into chunks which are converted into vectors with the help of embeddings performed on the received sequence. The generated vectors are stored in a vector database. This database is accessed when a similarity search is needed to be performed to determine the relevant documents for the input query provided by the user. The model uses the retrieved document and converts it into tokens which are transformed into vectors with the help of input positional embedding. This helps map each token generated from the text sequence to a higher dimensional vector which captures the semantic information of the token. The vectors are provided to the attention mechanism where the self-attention layer computes the relationship present between the different components of the input sequence converted into vectors. It contains key, value, and query computations that take the input vectors and transform them into three different representations where the weighted sum of the value vector is calculated based on

the similarity present between the query vector and the key vector representations. Due to the presence of a multi-head attention mechanism, the process of determining the relationships present between the different components of input can be performed parallelly and the output from all the computations can be combined. The output generated from the multi-head self-attention mechanism is provided to the feed-forward neural network that transforms the representations into the final output. The activations present in the feed-forward network undergo layer normalization which is added with the residual block to bring in more stability during the training process and prevent the model from getting overfitted. The stacking of multiple layers of multi-head self-attention mechanism and the feed-forward neural network is performed and the output generated from the final feed-forward neural network is provided to a linear layer added with softmax function which calculates the probability of each token and the one with the highest probability is selected as part of output sequence. This process continues until the final response is generated which is provided to the user.

Figure 52

Architecture and Data Flow Involving OpenAI GPT-4 Large Language Model



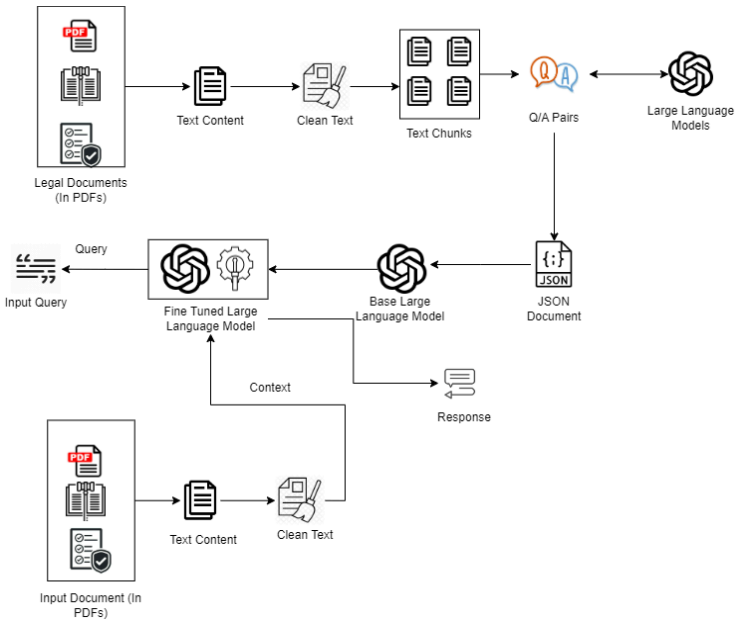
Note. The complete model architecture and dataflow involves OpenAI's GPT-4o LLM. Original from Team 1.

Fine-Tuning GPT-4o Model

Figure 53 depicts the workflow for fine-tuning the OpenAI model. Obtaining legal documents is the first step in the process. These include PDFs with past court cases brought against tech businesses for breaking regulations, corporate guidelines, laws pertaining to technology, and user agreements. Relevant text is extracted from these documents and then carefully cleaned and modified to highlight key legal facts. From this cleansed data, question-answer pairs are generated using OpenAI's native dashboard. These pairings are uploaded to the OpenAI model for fine-tuning after being saved in a JSONL format. In the process, the parameters of the model are fine-tuned to improve its capacity to produce precise and contextually relevant legal answers to user inquiries, rendering it a highly efficient tool for legal applications.

Figure 53

Workflow diagram involving fine-tuning of the model



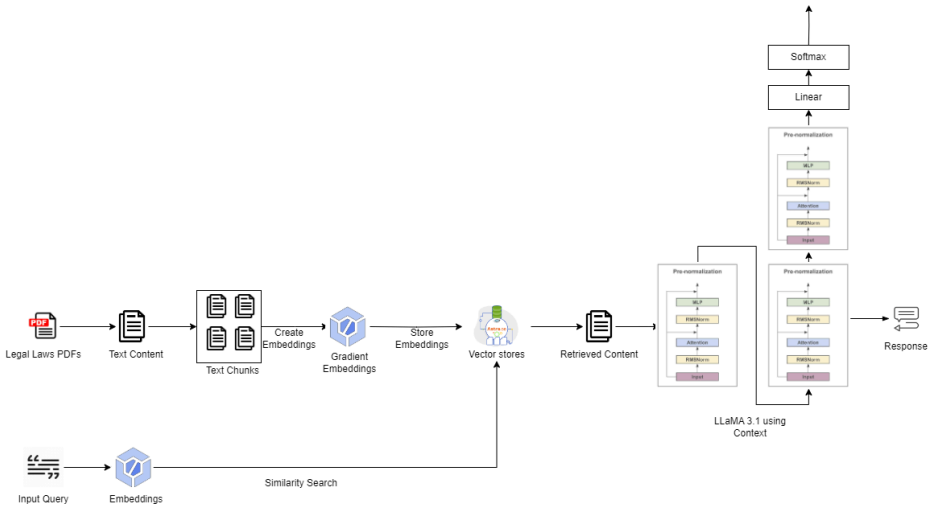
Note. Illustrating the steps involved in the workflow process where fine-tuning of the GPT-4o model is done. Original from Team 1.

RAG-based Llama 3.1 Model

Figure 54 shows the data flow which involves Llama 3.1 large language model. Similar to the data flow in Figure 52, it also uses the legal laws pdf documents that are pre-processed and converted into small chunks transformed into vectors, and stored in the Astra vector database. The similarity search is performed between the input vector (generated from a query provided by the user) and the vectors stored in the database to retrieve the set of documents or textual information that is most relevant. The textual content in it is divided into chunks which are further subdivided into tokens and transformed into vectors with the help of input embedding performed using the technique of RoPE. The normalization is performed before the self-attention layer using the mechanism of RMS layer normalization. Then the vectors are provided to the grouped query self-attention layer where the same operation is performed as GPT-4 and the output is provided to the feed-forward neural network where the SwiGLU activation function is used instead of ReLU. The final output generated from the last feed-forward neural network is provided to a linear layer with a softmax activation function that helps to calculate the probabilities of all tokens in the response. Out of this, the one with the highest probability was selected and added to the response query, which is provided to the user when completed.

Figure 54

Architecture and Dataflow involving LLaMA 3.1 Large Language Model



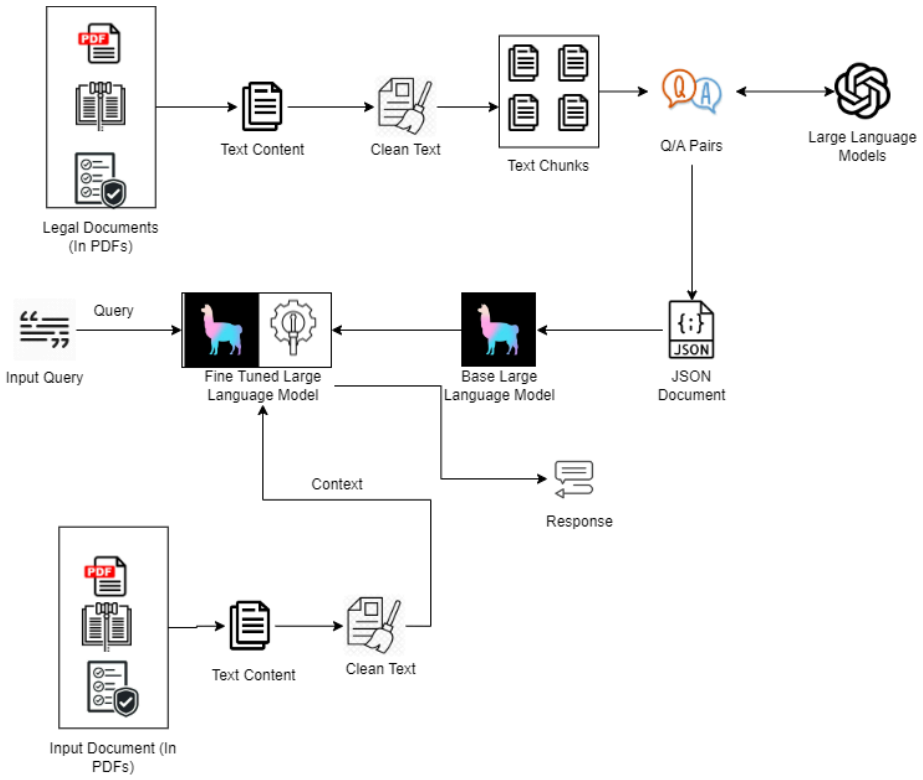
Note. The complete model architecture and dataflow involves Meta's LLaMA 3.1 LLM. Original from Team 1.

Fine-Tuning Llama 3.1 Model

Figure 55 shows the complete workflow that is followed to fine-tune the Llama 3.1 model where initially the legal documents consisting of the PDFs that contain historic legal cases filed against the tech-related companies for the violation of legal laws, policies that different companies adopt, legal laws that are related to technology, and terms of use and service that are defined by the companies which needed to be followed by every user to comply are processed to extract the textual content present in it which is further cleaned and transformed so that it contains appropriate information than the question-answer pairs are generated from it in which Open AI LLM is used. These question-answer pairs are stored in the JSONL file which is provided to the base model and its parameters are modified then the model is trained in such a way that it generates more appropriate legal-related responses to the user queries.

Figure 55

Workflow diagram involving fine-tuning of the model



Note. Illustrating the steps involved in the workflow process where fine-tuning of the LLaMa 3.1 model is done. Original from Team 1.

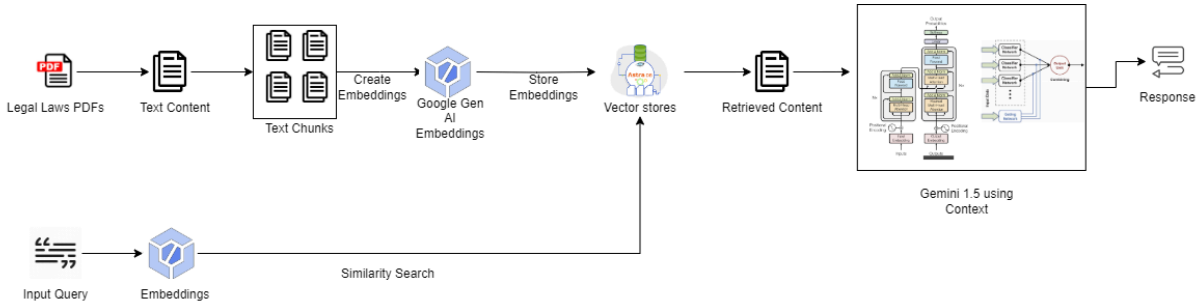
RAG-based Gemini 1.5 Model

In the case of the Gemini Model depicted in Figure 56, first, the PDF documents containing legal legislation are preprocessed and divided into manageable portions before being converted into vectors. Next, the Astra vector database contains these vectors. A similarity search is done between the input vector—which is produced based on a user-supplied query—and the vectors kept in the database during the retrieval procedure. The set of texts or documents that are most pertinent to the user's query are retrieved by this search. The text material that is extracted from the database is categorized into tokens and then into chunks. With the use of input embedding techniques like RoPE (Random Projection Encoding), these tokens

are converted into vectors. RMS layer normalization and other similar processes are used for normalization before the self-attention layer. After that, the vectors are passed into the grouped query self-attention layer, which carries out GPT-4o-like operations. The output of this layer is sent into a feed-forward neural network, which uses the activation function of a switched-gated linear unit (SwiGLU) rather than a recurrent unit (ReLU). The penultimate feed-forward neural network's final output is routed through a linear layer using a softmax activation function. The probabilities of each token in the answer are determined by this layer. After selecting and adding the token with the highest probability, the response inquiry is completed and given to the user.

Figure 56

Architecture and Dataflow Involving Gemini Large Language Model



Note. The complete model architecture and dataflow involving Google's Gemini 1.5 LLM. Original from Team 1.

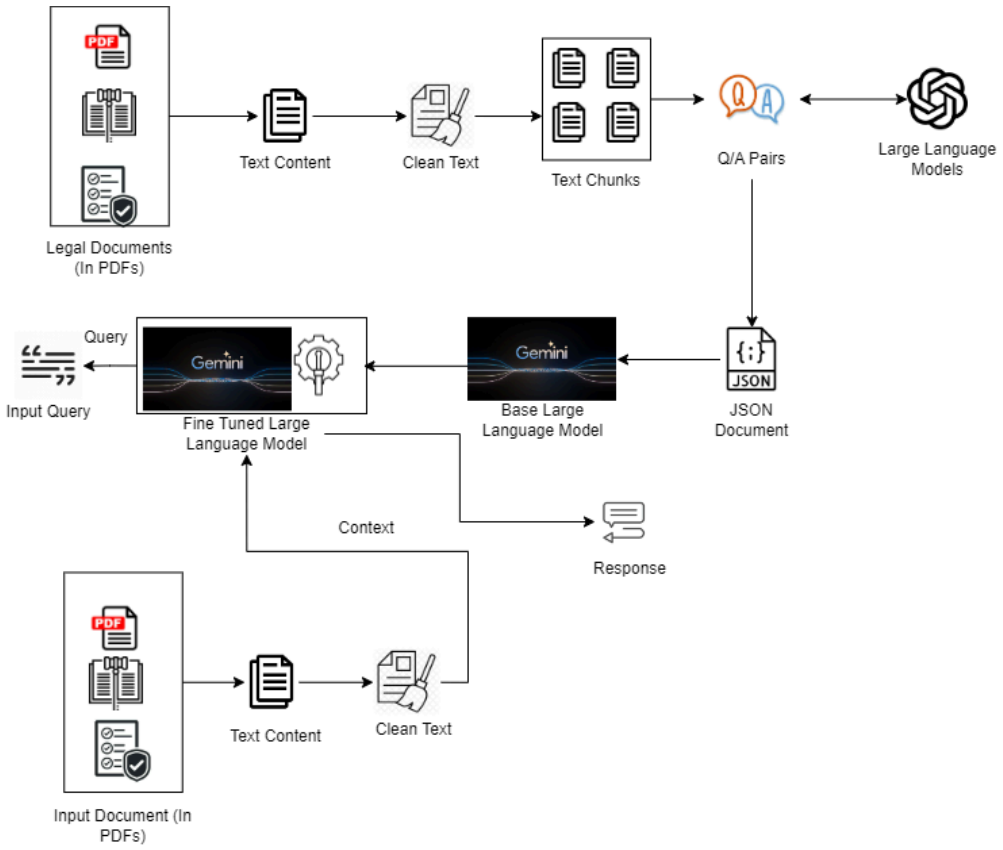
Fine-Tuning Gemini 1.5 Model

The procedure for fine-tuning the Gemini 1.5 model is shown in Figure 57. Legal documents are the first step in the process. These include PDFs of past court cases in which digital businesses were accused of breaking rules, company policies, legal frameworks pertaining to technology, and terms of service agreements that users must abide by. After the pertinent text from these documents is extracted, it is cleaned up and changed so that only the most crucial legal details are left. Question-answer pairings are created using Google AI Studio using this

cleaned data. The produced pairings are fed into the base Gemini 1.5 model for fine-tuning after being saved in a CSV file. In this stage, the parameters of the model are modified to guarantee that it can provide more accurate and legally sound responses to user queries, enhancing its effectiveness in legal contexts.

Figure 57

Workflow diagram involving fine-tuning of the model



Note. Illustrating the steps involved in the workflow process where fine-tuning of the Gemini 1.5 model is done. Original from Team 1.

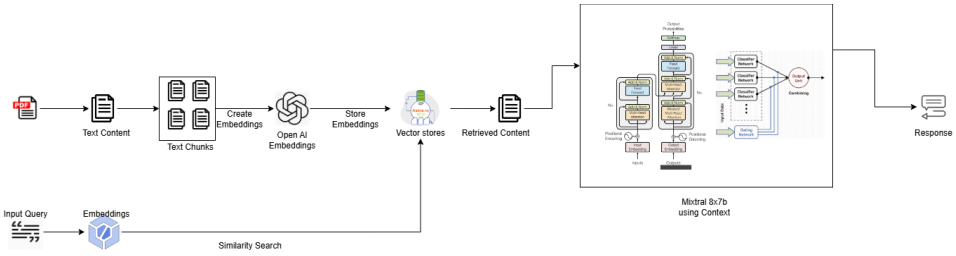
RAG-based Mixtral 8x7b Model

Figure 58 shows the data flow which involves the Mixtral 8x7b large language model. Similar to the data flow in Figure 6, it also utilizes the legal laws pdf documents that are pre-processed and divided into small manageable chunks, which are further transformed into

vectors using the concept of input positional embedding and stored in the Astra vector database. The vectors that are embedded are passed through the SMoE layer, where the router network makes use of the gating network mechanism paired up with the Top-K selection mechanism to calculate the scores for the selected experts. After processing the data through the SMoE layer the generated vectors are fed to the multi-head self-attention layer where the operation is similar to what is performed in the GPT-4 and the generated output is provided to the feed-forward neural network where the SwiGLU activation function is utilized. The residual connections and the process of layer normalization are performed on the generated output to increase stability and reduce the chance of overfitting. Then, similar to the previously discussed model the output generated from the last layer of the feed-forward neural network is provided to the linear layer where the activation function named Softmax is applied to calculate the probability score of each of the tokens in the generated response. The one with the highest probability is selected and included as part of the response. To help generate a relevant and meaningful response similarity search is performed between the input vector (generated from a query provided by the user) and the vectors stored in the database to retrieve the set of documents or textual information that is most relevant.

Figure 58

Architecture and Dataflow involving Mixtral 8x7b Large Language Model



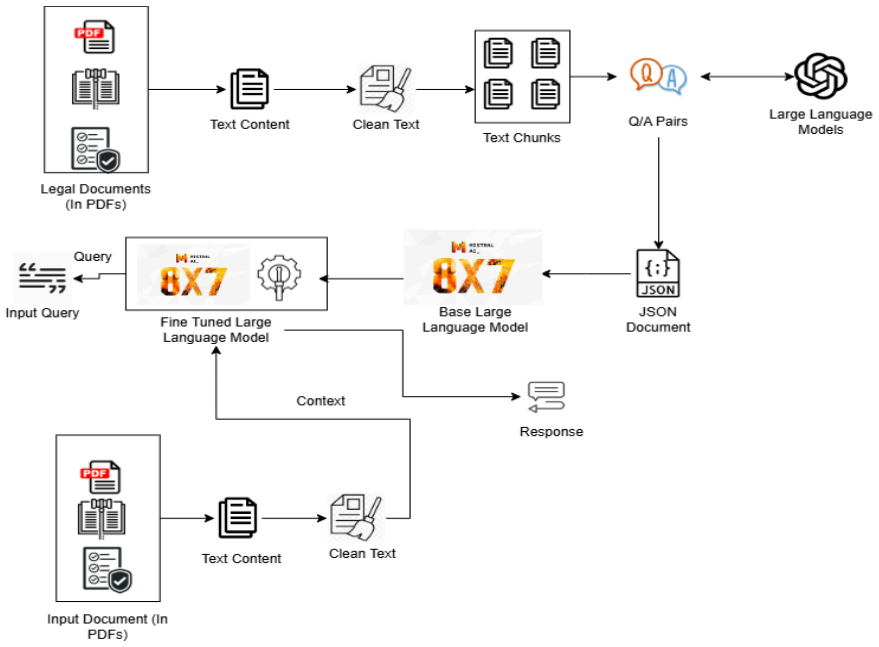
Note. The complete model architecture and dataflow involving Mistral AI's Mixtral 8x7b LLM. Original from Team 1.

Fine-Tuning Mixtral 8x7b Model

Figure 59 shows the complete workflow that is followed to fine-tune the Mixtral 8x7b model. The process initiates with the legal documents which are PDF files that contain information on the legal cases filed for the violation of the legal laws against the tech-related companies, policies adopted by various companies, and technology-related legal laws that are processed to extract the meaningful textual information which is further thoroughly cleaned and effectively transformed so that it contains properly structured and relevant information that could facilitate in generation of useful question-answer pairs using Open AI's LLM. The generated question-answer pairs are stored in the JSONL file, and this file is provided to the base Mixtral model where its parameters are modified to train the model in such a way that it generates more relevant responses to the legal-related user queries.

Figure 59

Workflow diagram involving fine-tuning of the model



Note. Illustrating the steps involved in the workflow process where fine-tuning of the Mixtral 8x7b model is done. Original from Team 1.

Model Comparison and Justification

Comparison

This section compares and justifies the models used in creating LexLLM, taking into account their strengths, limits, features, approaches, and targeted challenges.

Llama 3.1. Addresses intricate legal guidelines pertaining to FTC standards, COPPA, etc. Features include the ability to fine-tune and comprehend natural language. The method relies on large-scale language models (LLMs). Its strength in digesting legal texts and its capacity to comprehend complex legal frameworks are its strongest points. The computing resource intensity and possible scalability issues are among the limitations.

Retrieval Augmented Generation (RAG) is a hierarchical architecture used by Llama3.1 that combines retrieval and generating components. With this design, Llama3.1 can search through a vast corpus of legal papers for pertinent passages, and then it can produce responses based on the information it finds. Llama 3.1 improves its comprehension of legal vocabulary, structures, and contexts by fine-tuning legal datasets. This enables it to offer accurate and contextually relevant replies to legal inquiries. Because of its architecture, it can handle complicated legal requirements with ease, even when legal texts have a hierarchical structure and are lengthy.

OpenAI GPT 4o. Understanding and summarizing legal texts is the main focus of OpenAI GPT. Contextual awareness and natural language processing are among the features. The method relies on large-scale language models (LLMs). Strengths include adaptation to different domains and natural language processing capabilities. Limitations include potential context-related difficulties in highly specialized legal topics and biases in training data.

OpenAI GPT uses a transformer architecture to extract contextual information and long-range dependencies from text by relying on self-attention processes. Because of the way it is built, it can scan input sequences token by token, focusing on pertinent passages to provide responses that make sense and are acceptable for the given context. Because OpenAI GPT is trained on a wide variety of text material, including legal papers, it can generalize effectively across various domains. Despite lacking a dedicated architecture for processing legal texts, its extensive language comprehension skills enable it to efficiently interpret and summarize legal materials.

Gemini 1.5. Handles legal texts' intricacies. Context awareness and fine-grained text understanding are among the features. The method relies on large-scale language models (LLMs). Contextual awareness and a deep comprehension of legal texts are among one's strong points. Restrictions include possible scalability issues and restricted availability in comparison to other models.

Similar to OpenAI GPT, Gemini has a transformer-based architecture but adds improvements designed with legal text processing in mind. It can recognize complex legal intricacies and contextual clues because of its context-awareness modules and fine-grained text-understanding capabilities. Because of its architecture, Gemini can process legal papers more thoroughly, extract important information, and accurately identify pertinent legal concepts. Gemini outperforms more broad language models in the understanding and analysis of legal texts by incorporating domain-specific improvements into its architecture.

Mixtral 8x7b. Handles challenges related to the processing of comprehensive and hierarchically structured legal content that is commonly found in complex case laws and the standards established by COPPA, FTC guidelines, etc. The SMoE layer which is part of the

architecture helps to process a large amount of legal information by only activating a small number of parameters for each token. The unique architecture of Mixtral allows the model to focus on extracting relevant or key information and getting a grasp of the dependencies present in the content over the large sequence of input. This allows the model to identify the relevant section of the document that could be part of the response to the user query more appropriately.

Mixtral 8x7b has the capacity to process complex instructions efficiently which allows it to generate more precise and contextually relevant responses to user queries. It can easily be optimized to parse through the database storing legal information and generating the required information without utilizing a large amount of computational resources. Its hierarchical structure allows it to use parameters more effectively which helps it to reduce the inefficiencies observed in processing and improves the ability to get a much better understanding of the legal jargon. Table 14 depicts the comparison among different LLMs.

Table 14

Comparison of the four LLMs

Model	**Targeted Problems**	**Features**	**Strengths**	**Limitations**
Llama 3.1	Complex legal regulations governing COPPA, FTC standards, etc.	Natural language understanding, fine-tuning capabilities	Renowned for its robustness in legal text processing, ability to understand intricate legal frameworks, and fine-tuning capabilities for precise replies.	Computational resource-intensive, potential challenges in scalability, fine-tuning process may require significant expertise and computational power
OpenAI GPT 4o	Understanding and summarizing legal documents.	Natural language processing, contextual understanding	Widely recognized for its natural language processing capabilities, adaptability to various domains, and strong contextual understanding	They may exhibit biases present in training data, struggle with the context in highly specialized legal domains, and have limited contextual memory

Model	Targeted Problems	Features	Strengths	Limitations
Gemini 1.5	Addressing complexities in legal texts	Fine-grained text understanding, context-awareness	Notable for its fine-grained understanding of legal texts, contextual awareness, and ability to generate precise responses	Limited availability compared to other models, potential scalability challenges, may require extensive fine-tuning for optimal performance
Mixtral 8x7b	Processing comprehensive, and hierarchically organized legal content	Sparse Mixture of Experts (SMoE) based architecture, large context window, and context-aware	Effectively process large legal documents, strong ability to follow instructions, and ability to handle large amounts of context	Complex to train and deploy, fine-tuning is tough, and the absence of the build-in moderation mechanism to keep provided user queries in check

Note. The table summarizes the comparison of tables. Original from Team 1.

Justification

The choice of Llama 3.1 for handling the complex and dynamic terrain of legal rules is crucial because of its resilience in processing legal texts and its ability to be refined. LexLLM is well-founded due to its ability to comprehend and navigate the complexities of legal papers, especially those that control domains such as COPPA and FTC rules. With its widespread recognition and expertise in natural language processing, OpenAI GPT is an obvious choice for thoroughly comprehending and summarizing legal documents in a variety of fields. Because of its flexibility and adaptability, it is a priceless tool for guaranteeing that LexLLM can effectively handle a variety of legal questions and materials. Gemini's choice is essential for addressing the intricacies included in legal papers because of its in-depth comprehension of legal texts and

contextual awareness. LexLLM's power to provide accurate and pertinent responses is enhanced by its ability to comprehend complex legal intricacies and contextual clues. Mixtral's strong performance in processing large and hierarchically structured legal text due to its large context window and presence of SMoE layer make it a beneficial addition to LexLLM's arsenal, it can easily understand complex instructions which improves the ability to generate much more precise and relevant response. Table 15 below summarizes the justification for choosing these four LLMs.

Table 15

Justification for choosing the three LLMs

Model	Justification
Llama 3.1	Chosen because of its strong legal text processing and fine-tuning capabilities, which make it appropriate for handling intricate legal laws.
OpenAI GPT 4o	It was selected because of its well-known natural language processing powers, which enable it to comprehend and summarize legal papers in a variety of fields.
Gemini 1.5	Chosen due to its contextual knowledge and in-depth comprehension of legal texts, both of which are critical for handling the complexity found in legal documents.
Mixtral 8x7b	Chosen because of its ability to process hierarchically structured large-size documents effectively, and its capability to understand complex instructions to generate much more accurate legal response

Note. The table shows the justification for each dataset. Original from Team 1.

Model Evaluation Methods

This section contains the metrics and techniques of evaluation that were applied to each model used in the creation of LexLLM. Together with metrics from LangChain, we incorporate

conventional assessment measures that are frequently used for Large Language Models (LLMs). Every metric is well discussed, including its benefits and limitations as well as any relevant calculations.

Perplexity

The degree of perplexity in a language model's prediction of a sample text is measured. Better performance is indicated by less confusion. The corresponding formula is shown in Equation 13.

$$P = 2^{- \frac{1}{N} \sum_{i = 1}^{N} \log_2 P(w_i \mid w_{i - 1}, \ldots, w_{i - n + 1})} \quad (13)$$

Here, N is the number of words in the text, and P is the probability assigned by the language model to the next word given the previous n words.

Open AI Evals

OpenAI evaluations serve as crucial benchmarks for us to gauge the effectiveness of our LLM-powered PDF chat application. These evaluations provide invaluable insights into the accuracy and coherence of the text generated by our application. By analyzing the results of OpenAI evaluations, we can identify areas for improvement and refine the performance of our system to deliver more engaging and informative interactions. Utilizing OpenAI evaluations allows us to ensure that our PDF chat application meets the highest standards of quality and reliability, enhancing user satisfaction and confidence in its capabilities.

Human Evaluations

When evaluating how well a refined language model performs, human assessments are extremely important, particularly when the activity at hand involves complex comprehension, like responding to legal inquiries. Although automated criteria like accuracy or F1 score offer

quantitative insights, human evaluation provides a more comprehensive assessment of the model's capacity to produce coherent, relevant, and contextually appropriate responses. Experts can assess the model's reasoning quality, its capacity to understand challenging prompts accurately, and the output's clarity aspects that are challenging for algorithms to fully represent. This qualitative input is very helpful in pointing out the model's advantages and disadvantages, giving a comprehensive picture of how well it performs in practice, and directing future refinement efforts.

Model Validation and Evaluation Results

We compare perplexity (where lower values imply better model confidence) and OpenAI evaluation accuracy (where higher values denote better answer quality) to assess the performance of twelve different models on a set of one hundred Q&A pairs. The models, which are assessed in three configurations—Base, RAG, and Fine-Tuned—include Gemini 1.5, OpenAI GPT-4o, Mixtral 8x7b, and Llama 3.1.

While OpenAI evaluation accuracy directly evaluates how frequently the models deliver true responses based on the evaluation set, perplexity serves as a statistic to quantify how well each model predicts the likelihood of the correct answer. Figure 60 and Figure 61, respectively, show the outcomes.

To thoroughly evaluate model performance, 100 Q&A pairs representing a variety of real-world legal contexts were assembled to create the assessment set (sample shown in Figure 10). The responses produced by each model were limited to less than 100 tokens in order to maintain uniformity and enable an equitable evaluation of their capacity to deliver succinct and pertinent answers.

Figure 60

Sample of the evaluation set to evaluate LLM models

```
Q: What happens if I violate the terms of use for Microsoft Co-Pilot?
A: Violating the terms of use may result in suspension or termination of access to Microsoft Co-Pilot.

Q: Does Microsoft Co-Pilot have a user community?
A: Microsoft may have forums or user communities where users can share tips and ask questions about Co-
Pilot.

Q: Are there age restrictions for using Microsoft Co-Pilot?
A: Users must generally be at least 13 years old to use Microsoft Co-Pilot, in compliance with
applicable laws.

Q: Can I integrate Microsoft Co-Pilot with other software?
A: Microsoft Co-Pilot may offer integration options with other Microsoft and third-party software,
subject to compatibility.

Q: What is the liability of Microsoft regarding Co-Pilot?
A: Microsoft's liability may be limited as stated in the terms of use, usually excluding indirect or
consequential damages.
```

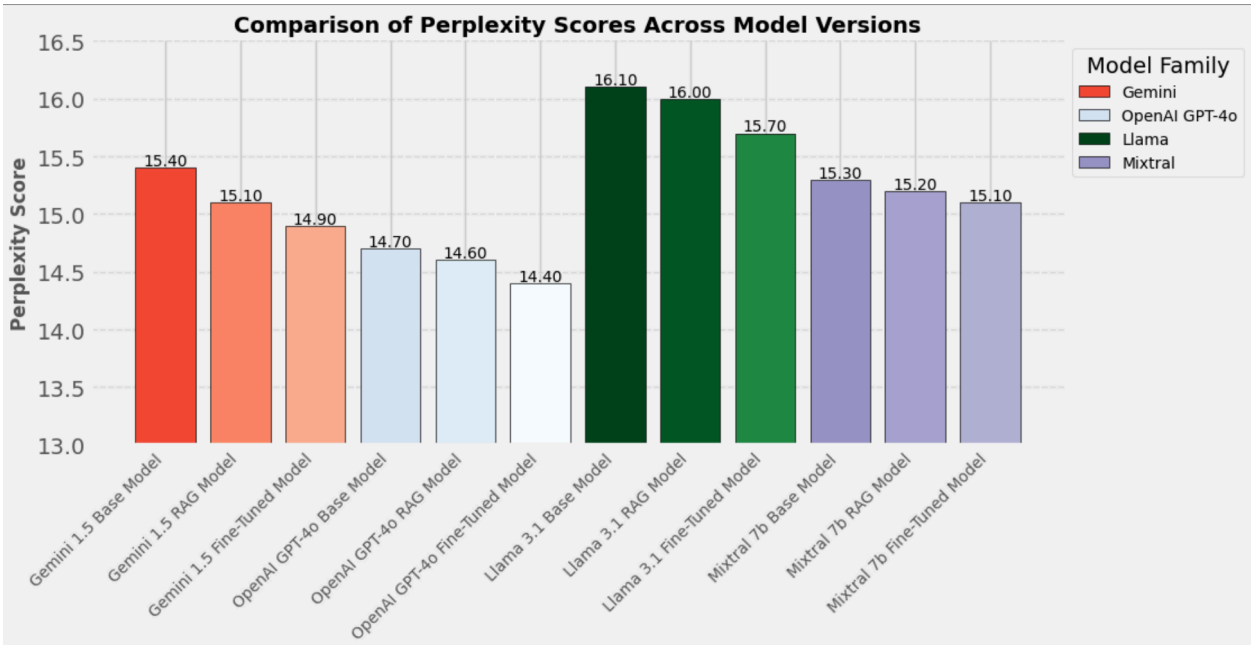
Note. A sample of the evaluation set was used for the evaluation of the twelve models. Original from Team 1.

Figure 61 illustrates how, for every model in each model family, the perplexity values show marginal gains as we go from Base to RAG and Fine-Tuned versions. As an example, the perplexity of the Gemini 1.5 Base model is 15.4. This increases slightly to 15.1 for the RAG version and even more to 14.9 for the Fine-Tuned model, which is a 3.2% improvement over the base model. In a similar vein, the OpenAI GPT-4o models perform better, with perplexity improving 2.2% overall from the Base to Fine-Tuned, from 13.7 in the Base model to 13.6 for RAG and 13.4 for the Fine-Tuned version. The Llama 3.1 models exhibit the least improvement, with perplexity decreasing by only 2.5% from 16.1 in the Base model to 16.0 for RAG and 15.7 for Fine-Tuned. The Mixtral 8x7b models showed the least improvement of 1.3% in perplexity by decreasing their value from 15.3 in Base to 15.2 in RAG and 15.1 in Fine-Tuned. This implies that despite being slightly superior to their Base and RAG counterparts, the Fine-Tuned models did not demonstrate appreciable decreases in confusion because of the small amount of extra

training they received. However, the OpenAI GPT-4o Fine-Tuned model consistently scored the lowest perplexity (13.4), demonstrating its higher confidence in producing answers with the least amount of uncertainty. Perplexity improved from Base to Fine-Tuned.

Figure 61

Perplexity score of each model



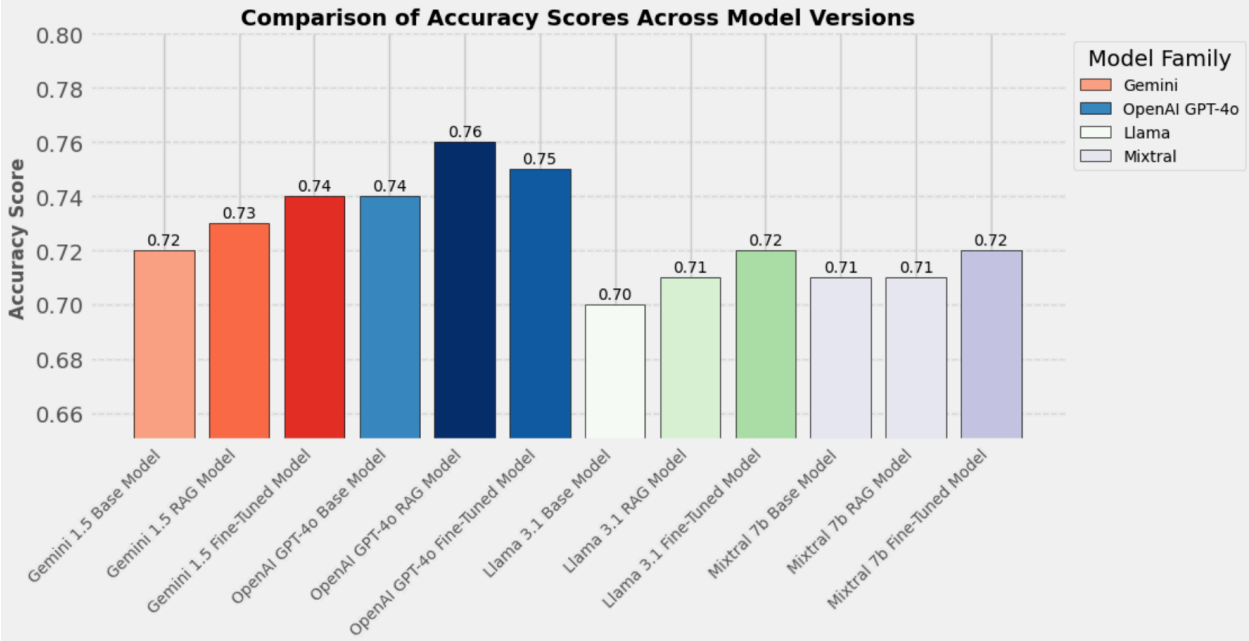
Note. Bar chart showing the perplexity scores of all models on the evaluation set. Original from Team 1.

However, the OpenAI evaluation accuracy findings are shown in Figure 62, where greater numbers denote better performance. The accuracy of the Gemini 1.5 Base model is initially 72%; it then slightly improves to 73% in the RAG version and 74% in the Fine-Tuned version, indicating a 2.8% gain over the Base model. But out of all the models, the OpenAI GPT-4o Fine-Tuned model performs best, with an accuracy of 76%, 2.7% higher than its Base accuracy of 74%. The Base model in the Llama 3.1 family starts at 70% and rises to 71% for RAG and 72% for the Fine-Tuned model, which is a slight 2.9% improvement. The Mixtral 8x7b

Base model shows the minimum accuracy among the three configurations with a value of 71%, which remained the same in the RAG model with a value of 71% and increased in the Fine-Tuned model to 72%, this portrays a very small increase of 1.4%.

Figure 62

Accuracy score of each model



Note. Bar chart showing the OpenAI Eval accuracy scores of all models on the evaluation set. Original from Team 1.

These findings unequivocally demonstrate that although the Fine-Tuned models offer increased accuracy, the gains are only marginally larger because of the restricted amount of additional training. The top-performing RAG model (GPT-4o RAG) only achieves a 2% accuracy gain over the Base model, demonstrating the negligible performance differences between the RAG and Base models.

In terms of producing responses that are both confident and accurate, the OpenAI GPT-4o fine-tuned model is the most effective overall. It has the lowest perplexity and the highest

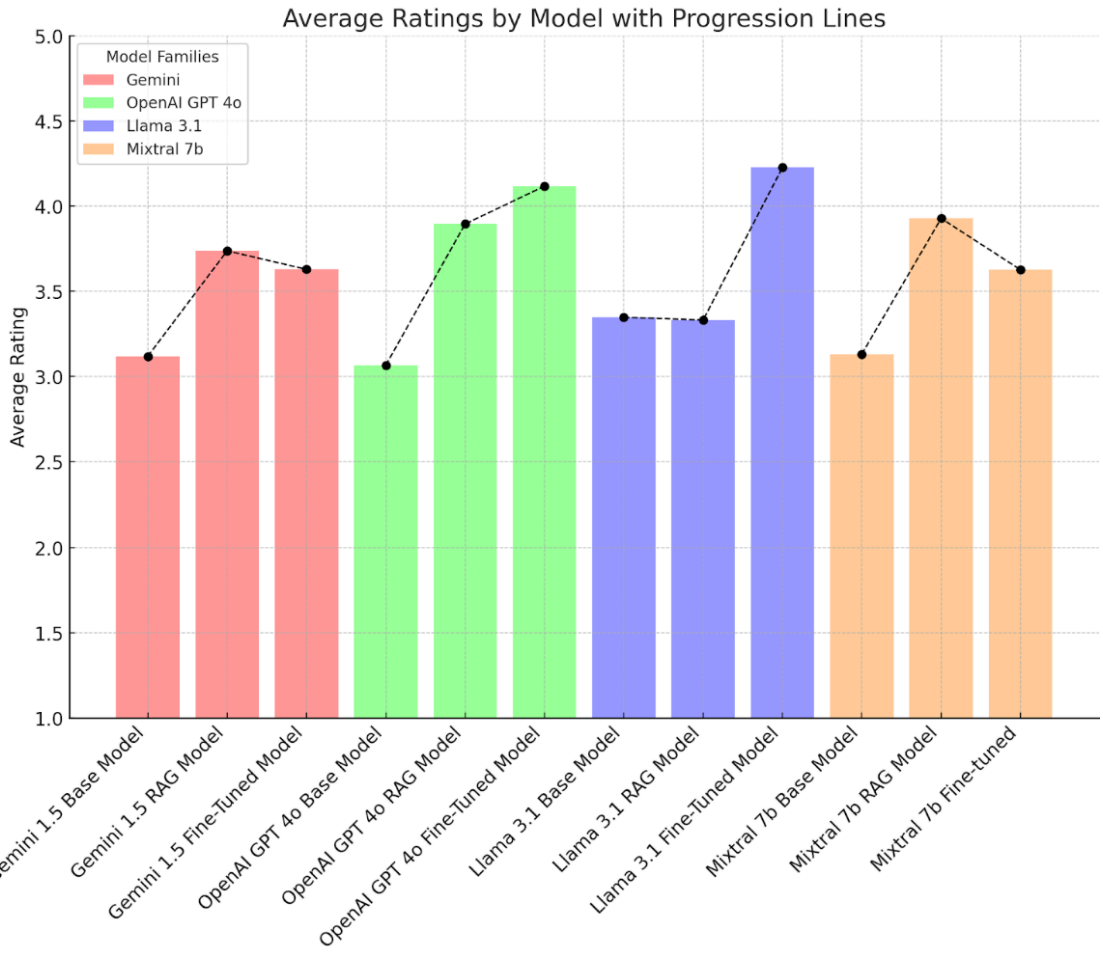
evaluation accuracy. With gains of 2-3% in both perplexity and accuracy, the models that are part of the Gemini, and Llama families show more subdued advances. The least amount of advancement was seen in models of the Mistral family where the gain in perplexity is around 2% but gain in accuracy is only 1%. This is in line with the minimal training given to their RAG and Fine-Tuned counterparts.

Human Evaluation

76 replies were gathered over the course of 12 days for the human evaluation, which evaluated 12 legal AI chatbot models in three categories: Base, Retrieval-Augmented Generation (RAG), and Fine-Tuned. Participants assessed models on a scale of 1 to 5, reflecting three levels of skill (28% Beginner, 45% Intermediate, and 27% Advanced). With Fine-Tuned models reaching a peak average of 4.2 in the OpenAI GPT 4o family and Base models averaging 3.1, the average rating for all models was 3.4. The Gemini family showed notable performance improvements, with the RAG model outperforming its Base counterpart by 19% and the Fine-Tuned versions by 15%. Comparing Mixtral 7b RAG models to their Base equivalent, the former obtained 3.9, a 25% gain. Figure 63 shows the average approval rating for the models after human evaluation.

Figure 63

Average rating of models after human evaluations



Note. Bar chart showing the average ratings of all models in the human evaluation. Original from Team 1.

Data Analytics and Intelligent System

System Requirements Analysis

System Boundaries, Actors and Use Cases

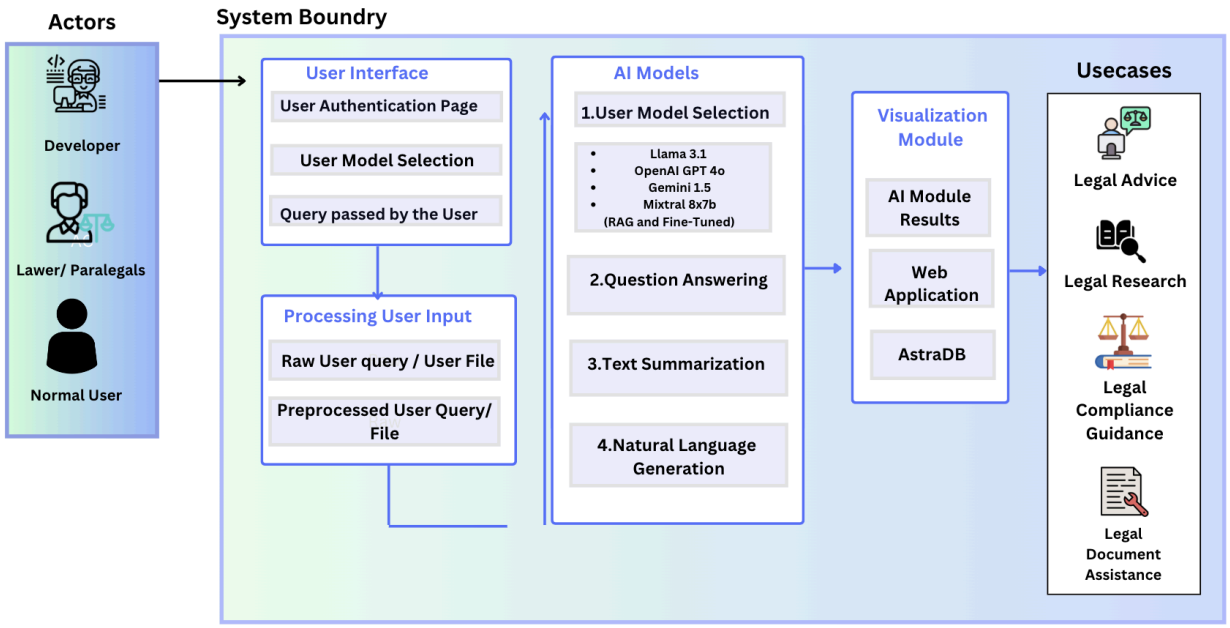
The LexLLM application is built to efficiently answer questions related to legal documents with the aim of comprehending complex legislative code documents. With the extensive functionality of the application in place, the system's boundary includes the queries

submitted by the user and the language models generate the relevant response to the questions based on the knowledge as well as inputs such as files shared by the user. The application users can be lawyers, paralegals, the general public, or anybody who wants to research or explore legal documents. The user inputs the query along with the model selected by the user, and the input is passed to the AI system to generate an answer. The system is capable of interacting with users in a human-like conversation manner. To provide better and more accurate answers, the AI model is trained utilizing a large quantity of legal data and powerful GPUs. the users can ask any time of question such as open-ended questions as well as specific questions relating to a law-related document passed to the model by the user. Based on the model selected by the user, the response is generated accordingly. The chat history of a specific session is maintained as the user can refer back to the conversation at any point in time. Figure 64 illustrates the different actors involved, the system boundary, and the use cases of the application. The following lists some of the use cases of the application:

1. Legal Advice: LexLLM is a legal assistance for any individual who wants to understand the basic understanding of the law and legal documents based on existing legal cases.
2. Legal Research: The user researching the legal sector can make use of the application to understand the case law, statutes, or legal precedents based on particular legal questions or topics.
3. Legal Compliance Guidance: The proposed system provides support with compliance advice like Corporate law, Company Constitution, Corporate Liability, etc.
4. Legal Document Assistance: LexLLM is built to accept documents from the user, parse the document, and answer any questions related to the document.

Figure 64

Architecture showing Actors, System Boundary, and Use Cases



Note. A schematic illustrating the use cases, actors on the system, and system boundary. Original from Team 1.

Data analytics and Machine Learning capabilities

The suggested approach deals with a wide range of intricate legal frameworks that are challenging for the common citizens to comprehend, such as legislative code papers, and to stay up to date with the frequent alterations. The application incorporates numerous data analytics and machine-learning techniques. Utilizing the diverse amount of legal documents, legal laws, and organizational laws the preprocessing steps involved various Natural Language Techniques (NLP) involving data annotation and augmentation to make the data ready for model understanding. Below is the list of machine learning capabilities of the system:

1. Natural Language Understanding: LexLLM can understand the human natural language using large language models trained on humongous web content, it can build a

human-like conversation. The input queries are tokenized, normalized, and undergo word embedding, and contextual understanding to generate relevant answers to user queries.

2. Natural Language Generation**:** LexLLM can generate and interpret the question and documents shared by the user to generate responses.Also, further, explain in detail the answers generated by the system.
3. Text Summarization**:** The system is capable of generating summaries of the document shared by the user by parsing and understanding the document to generate the response. The system can handle long texts and be able to generate quick summaries of the content shared.
4. Question Answering: LexLLM has the capability to answer any questions as it has been trained on large volumes of legal data. Based on the training the system is able to generate factual answers to the questions submitted by the user.

System Design

System Architecture and Infrastructure

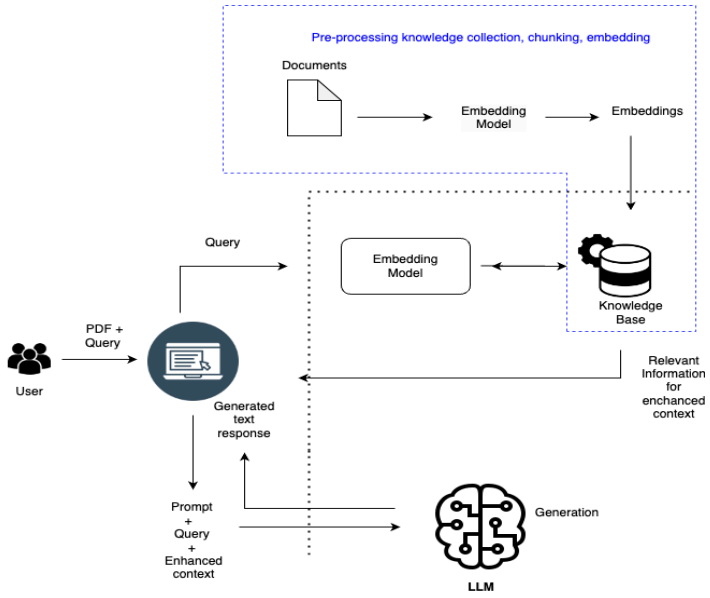
The proposed system architecture consists of the user interface, Backend, and Cloud. Figure 65 emphasizes key elements and how well they work together. The user interface is the front end where the user interacts with the system by uploading a PDF of the legislative sector, selecting from the available LLM models to process their queries regarding the uploaded document.

The backend layer is the layer where user requests are handled and replies are produced. This layer includes essential components: (1) Processing PDF where the texts are extracted and pre-processed (2) Vector database and (3) RAG implementation of three different LLM models (LLaMa 3.1, Gemini 1.5, GPT-4o, Mixtral 8x7b). Q&A pairs are generated and saved in CSV

and JSON format to fine-tune these models further. The cloud infrastructure ensures data storage protection.

Figure 65

Basic System Design



Note. General RAG-based system architecture. Original from Team 1.

System Supporting Platforms and Cloud Environment

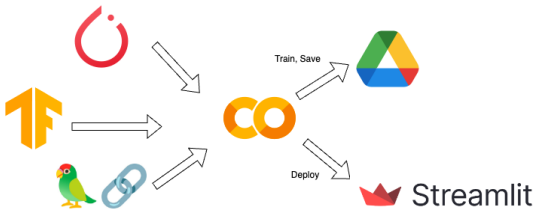
Figure 66 illustrates how the frameworks, cloud environment, and supporting platforms of LLM Chat application provide a modular ecosystem designed to facilitate training, development, and deployment. Google Drive is the central repository for storing PDF datasets. These platforms provide effective dataset management and organization by providing scalable and secure storage options. Google Drive enables team members working on data preparation, annotation, and model building to collaborate more easily with helpful features like version control and file sharing. The system leverages the main frameworks for model building are tensorflow, pytorch, langchain. To benefit from Google Colab's GPU-accelerated features and collaborative capabilities, these frameworks are integrated into it. Google Colab serves as the

main center for storing training data and training models. The system also integrates with external resources such as OpenAI API for the generation of question-and-answer pairs. When the models are ready for use, they are integrated by the StreamLit application, which provides a user interface to communicate with the AI system.

Overall, the design of the system makes use of machine learning frameworks, local computing resources, and local storage solutions like Google Drive to offer scalable, dependable, and effective RAG-based PDF chat applications.

Figure 66

Supporting Systems and Environments



Note. Supporting platforms, frameworks, and a cloud environment. Original from Team 1.

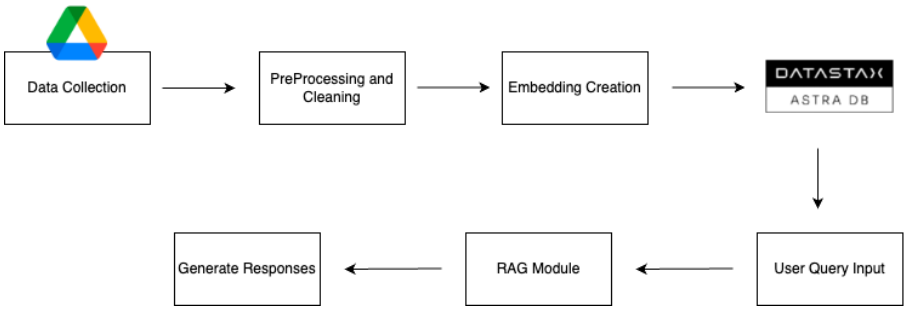
System Data Management Solution

Figure 67 demonstrates the data flow of how the LexLLM system handles user interactions with prompts related to the legislative sector, supported by a detailed data management plan. Data management solutions can entail various tasks, including collecting, cleaning, storing, and monitoring data in addition to ensuring the data is well-governed. The PDF datasets are sourced from various platforms. These raw PDFs are categorized and stored in three different folders in Google Drive, ensuring easy access and effective storage administration. Once collected, these raw PDFs further undergo preprocessing to format them for model training. Once the data is in the desirable format, vector embeddings are created and stored in a vector database. In order to produce contextually appropriate responses to these prompts, the backend

incorporates the RAG technique. The RAG module in the backend uses Astra DB as the vector database to fetch the responses for the user query. In addition, the data management plan includes scalability to handle future growth in data volume and additional data sources and mechanisms for handling errors during preprocessing. Along with this, the system ensures clear roles, responsibility, and ownership, establishing granular access control for data usage. This strategy ensures that the data is well-governed and ready for future improvement.

Figure 67

Data Management Design



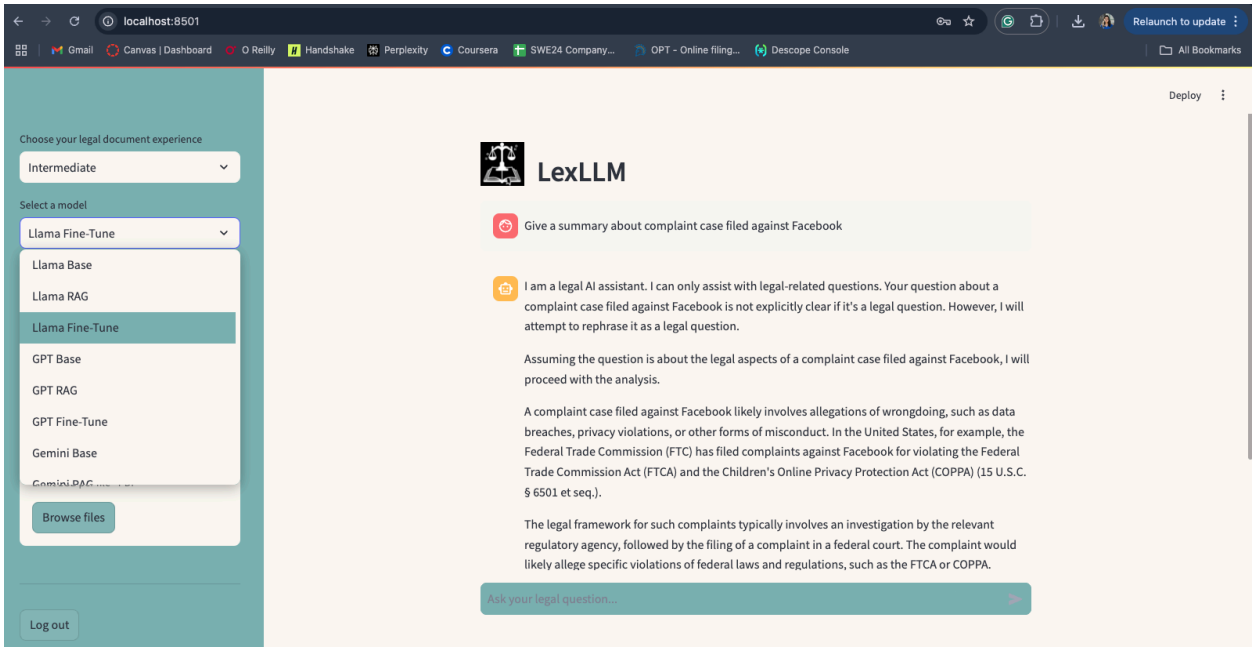
Note. System Data Management Solution. Original from Team 1.

System User Interface

Figure 68 shows the example interface of LexLLM. The "LexLLM," the name of the user interface, is kept simple, and other functionality will be added in the future. Users can choose a model from a dropdown that includes "Llama 3.1", “Gemini 1.5”, “GPT-4o”, and “Mixtral 8x7b” and choose a legal document experience from a menu on the left. A "Browse files" button allows you to upload files beneath the model selection. There is a text box for user input at the bottom titled "Have a Question?" where the user can write their query. Once the query is posted by the user, a response output is generated by the chosen LLM model. The general arrangement is simple and easy to use.

Figure 68

User Interface



Note. System User Interface. Original from Team 1.

Intelligent Solution

AI and Machine Learning Development

The Lex-LLM AI Legal Assistant is designed to address the complexities of legal document analysis, which allows users to navigate legal frameworks that are intricate. To understand complicated legal documents and effortlessly answer legal inquiries we developed a Retrieval-Augmented Generation (RAG) system. Four state-of-the-art models, LLaMa 3.1, Gemini 1.5, GPT-4o, and Mixtral 8x7b are integrated into this solution. Each model is fine-tuned to boost its performance in generating contextually appropriate responses to user inquiries and comprehending legal texts.

GPT-4o relatively does well at generating legal narratives with human-like articulation, including summaries of court decisions, legal opinions, and answers to challenging legal issues.

It leverages a transformer-based architecture with multi-layer attention mechanisms and position-wise feed-forward networks. It also uses autoregressive decoding to predict the next word based on contextual input, assuring coherent and legally sound responses. GPT-4o employs maximum likelihood estimation(MLE) to maximize next-word prediction accuracy in legal situations. It is an effective model for producing high-quality legal summaries and opinions because of its training, which incorporates billions of tokens. This allows it to generate concise legal writings appropriate for quick insights or in-depth study.

Gemini 1.5 is built to efficiently handle complicated legal reasoning and multi-turn legal debates. It is based on a Mixture of Experts (MoE) design, that can effectively handle complex conversations on case law, rules, and legal compliance by dynamically allocating computational resources during inference. Its architecture makes it possible for it to preserve context across multiple encounters, guaranteeing that the responses are rich in information and pertinent to the given context. The model ensures clarity and correctness in responses to questions about legal or regulatory compliance by employing its capacity to maintain consistency throughout prolonged legal debates. Gemini 1.5 also improves user interactions by including sophisticated conversational capabilities and offering insightful, legally sound responses that are tailored to the peculiarities of the legal domain.

LLaMa 3.1 is employed due to its advanced performance in long-context processing, which is an essential attribute for comprehending lengthy legal documents. The model's advanced self-attention mechanisms allow it to read legal papers with many clauses accurately and efficiently while preserving contextual integrity, even across long stretches. LLaMa 3.1, refined on domain-specific datasets, is highly effective at extracting and condensing relevant knowledge from these intricate texts, guaranteeing that the produced responses maintain legal

accuracy and coherence. It plays a critical role in precisely responding to complex legal questions by using its advantages in long-context comprehension.

Mixtral 8x7b is an important component of the RAG system for managing convoluted document parsing and high-precision legal data retrieval because of its impressive ability to read dense and organized legal material. With its multi-layered transformer architecture designed for processing large amounts of data, Mixtral 8x7b excels in extracting very fine details from contracts, statutes, and procedural documents. Cross-attention methods are used in its encoder-decoder architecture to preserve fine-grained context, assuring that the model reliably gathers and retrieves content. To be able to provide responses that are accurate and in line with specific legal terminologies and nuances, Mixtral 8x7b also uses an ensemble approach, integrating insights from multiple domain-specific submodels. Because of this model's focus on high fidelity and legal correctness, Mixtral 8x7b is dependable for Lex-LLM users.

Input and Output Requirements, Supporting Systems, and Solution APIs

Input Datasets. A broad range of legal documents are used as input datasets by the LexLLM AI Legal Assistant. These consist of legal statutes, company policies, court case PDFs, and law-based textbooks which undergo a rigorous preprocessing phase that includes data extraction, cleaning, and transformation using Python tools such as PyPDF2, NLTK, and Regex. To prepare the dataset for embedding and additional processing, preprocessing techniques include lemmatizing words, tokenizing phrases, and eliminating noise. After cleaning, the data is divided into smaller chunks to fit within token size constraints, guaranteeing effective vectorization and input for the model. These preprocessed texts and question-answer pairs are

generated to produce fine-tuning datasets tailored to each model (GPT-4o, Gemini 1.5, LLaMa 3.1, and Mixtral 8x7b).

Expected Outputs. The system is engineered to produce a variety of outputs and offers contextually relevant answers to user inquiries. Among these outputs are thorough legal summaries, which provide summaries of legal papers and highlight important ideas from extensive texts. Compliance recommendations are part of the expected outcomes where actionable advice is provided which are based on risk assessments for possible legal problems and requirements. It is also expected to generate legal opinions where each model specializes in a certain area. For example, GPT-4o produces legal opinions that are clear and simple, while LLaMa 3.1 is excellent at managing and summarizing large legal documents. Multi-turn legal dialogues are the specialty of Gemini 1.5, which enables interactive answers to follow-up inquiries. Mixtral 8x7b assures that extremely accurate, context-specific information is obtained by the users from structured legal texts by excelling in offering precise phrase extraction and thorough comprehension of legal statutes. The intention is to provide consumers with factual and legally reliable data.

Supporting System Contexts. The project stores and manages vector embeddings using Astra DB, a cloud-based vector database. Model-specific APIs are used to construct embeddings: Grow for LLaMa 3.1 and Mixtral 8x7b, OpenAI for GPT-4o, and Google Generative AI for Gemini 1.5. These make it possible to retrieve relevant documents through effective vector similarity searches. For fine-tuning, GPT-4o, LLaMa 3.1, and Mixtral 8x7b use OpenAI's LLM to produce question-answer pairs and Gemini 1.5 uses Google AI Studio. For scalability, the system needs high-performance GPUs and cloud infrastructure; code development and testing are done using Google Colab.

Solution APIs. The LexLLM AI Legal Assistant uses some important APIs. By employing the OpenAI API, GPT-4o generates summaries, human-like responses to complex queries, and the highest caliber legal viewpoints. The Gemini API allows Gemini 1.5 to deal with complex legal reasoning and multi-turn debates, guaranteeing coherence in prolonged discussions. To enable effective long-context legal document processing, we also plan on using an API for LLaMa 3.1 and Mixtral 8x7b enabling extraction and summarization of important information from lengthy legal documents. The interface is expected to provide smooth interactions between models and offer precise query handling, document retrieval, and response generation by integrating these APIs. Question-answer pairings are generated using Open AI and Google AI Studio APIs.

System Supporting Environment

Various technologies, platforms, and frameworks are used during the project's development. They supported the operations needed to create an AI-powered LLM-based Legal Assistant that helps to respond to different user queries related to the legal domain which will aid in enhancing the knowledge of the general public as well as help the people in the legal profession to get information about the laws or policies in concise form which can help them to make effective decisions. Among the several resources most commonly used is Google Collab. The source code of different operations involved at numerous stages of the project is written and run on it. All the data involved, either the raw one or the one obtained after processing are stored in Google Drive. It is the main resource used for storage. Several other resources are used to ensure the successful operation of the project.

Google Colab

It is a cloud-based environment that allows a facility like Jupyter Notebook where the users can easily write and execute a single line or a block of Python code directly into the web browser. It is well suited for exploring and processing the data, performing model training and development, and creating interesting visualizations. It is used in this project because of the computational power it possesses. It provides the facility to use GPUs and TPUs at no cost which is useful for performing the complex operations of the project like carrying out fine-tuning of different LLMs but at the same time keeping the budget in check.

Google Drive

It is a cloud-based central repository that provides the service of storing, accessing, and sharing files of different formats with any device having the proper connectivity to the internet. It is useful for storing various PDF files, Colab Notebooks, processed text files, model checkpoints, and documents for the project. It allows multiple users or team members to access the same file in real-time which is helpful when the team members are not working on a single device or location as it enhances collaboration and it also maintains a proper track of updates performed on different files which ensures proper management of versions. It is the main storage component of the project as all the resources are stored in it and it is usually accessed to retrieve files and content used to perform different tasks at different stages of the project.

GitHub

It is a web-based application or platform that provides the capability to work collaboratively by allowing multiple users or team members to upload or modify different files used in the project. It properly maintains the changes performed either on the Python scripts or some documents uploaded on it by properly enforcing version control. It proved helpful for the

project to keep track of the modifications made on different documents. It also acts as the web-based storage of useful resources and documents that any user can access in the future to get an idea about the complete project and how the different code files are useful for different phases of the project.

Streamlit

It is the Python framework which is open source and it is commonly used to create interactive and useful web applications within a short period of time. It proves useful for developers and data enthusiasts having little to no expertise in web development as they can easily convert their Python scripts into the properly functioning web application. It is used in the project to create the User Interface where the users can enter the prompts or queries related to the legal domain and the LLMs will generate a suitable response. It is useful for non-technical users (like Lawyers, paralegals, and the general public) to easily interact with different LLMs and receive the responses without being required to have much knowledge about programming.

LangChain

It is one of the most powerful frameworks used in the project as it enhances the abilities of the LLMs to generate the context-aware (legal domain) response by integrating it with specific domain data and utilizing various computational tools. This allows the LLMs to perform effective data retrieval and improve the capability to generate logical responses. In the project, this framework is used to generate responses by initially retrieving the relevant pieces of information, and for that Retrieval Augmented Generation (RAG) is performed. When the user provides the prompt the RAG retrieves the relevant information and then it is used to generate the final response. This ensures that the response is more related to the query and much closer to what was expected.

Draw.io

It is a web-based tool that allows users to create interesting flowcharts, diagrams, etc at completely free of cost. It provides a feature where the user can easily drag and drop or just select the icons required and can easily modify them. In the project, this tool helped to create project flow and various data flow diagrams.

Canva

It is a user-friendly, easy-to-adapt graphic designing platform that is used to create great presentations and logos. It is easy to interact and navigate which allows even the beginners to work without any issues. In the project, it is used to create interactive presentations. This tool also enhances collaboration as multiple team members can access and add/modify its content at the same time.

System Evaluation and Visualization

Analysis of Model Execution and Evaluation Results

Perplexity and OpenAI Eval Accuracy are the two main metrics we use in this project report to assess how well different language models perform when responding to a varied set of 100 Q&A pairs. OpenAI Eval Accuracy measures the percentage of right answers; higher numbers indicate better accuracy, whereas Perplexity measures model confidence; lower values indicate better prediction certainty. Three configurations of the models—Base, RAG, and Fine-Tuned for Gemini 1.5, OpenAI GPT-4o, and Llama 3.1 and Mixtral 8x7b—were examined, as illustrated in Figures 13 and 14. With the lowest perplexity (13.4) and the highest OpenAI Eval Accuracy (76%), the Fine-Tuned GPT-4o model outperformed all other models, demonstrating its resilience in terms of both accuracy and confidence.

When the evolution within each model family was examined, the Gemini 1.5 models showed significant gains, with accuracy rising by 2.8% and perplexity decreasing by 14.9% from Base to Fine-Tuned (15.4 to 11.2). The Fine-Tuned version of the Llama models only achieved a 2.5% reduction in perplexity and a 2.6% improvement in accuracy over the Base model, highlighting the relatively modest training effects of the Llama models. While the GPT-4o family demonstrated the most notable accuracy gains across configurations, particularly with the RAG and Fine-Tuned versions, achieving an increase of 5.3% in accuracy over the Base configuration, the Gemini Fine-Tuned model showed the largest reduction in perplexity of any model. Notably, the OpenAI Eval Accuracy of the GPT-4o RAG and Fine-Tuned models was consistently high.

Following these findings, we used the Mistral 7B model to conduct initial testing, assessing both the Base and Fine-Tuned versions. According to preliminary results, there was no discernible change in accuracy or perplexity between the Base and Fine-Tuned models, indicating that there would be no further value to fine-tuning this model. In subsequent iterations of this project, additional assessments will be carried out to validate these findings and determine whether Mistral 7B is suitable for more intricate Q&A jobs.

In addition to offering useful benchmarks for future enhancements, such as possible modifications to training intensity and model design, these comprehensive metrics enable a comprehensive assessment of the model's efficacy.

Achievements and Constraints

Achievements

Collected Legal Documents from multiple sources and created Question Answer pairs. The research process involved collecting legal documents consisting of Legal Laws, Legal cases, and Policies. These papers posed a serious challenge because of their diverse

formats—including different table of contents structures, image placements, layout designs, and the usage of special characters. Due to this variability, a thorough preprocessing step was required to normalize the data for analysis. After an intensive preprocessing step, the data was normalized to be used for question-answer pair generation.

Table 16

Data Statistics

Documents	Count of Files
Total Legal Cases PDFs	631
Total Legal Laws PDFs	2994
Total Policies PDFs	6611
Total File	10,236

Note. The table shows the number of documents collected for each category. Original from Team 1.

As shown in the above table 16, we have collected 10,236 documents in total consisting of 631legal cases, 2994 legal laws, and 6611 Policies.

Implemented 4 Fine-Tuned Large Language Model and with RAG. The ultimate goal of this project is to develop a sophisticated Legal Assistant designed to make legal information more accessible to the general public. To achieve this goal, cutting-edge large language models (LLMs) such as Llama 3.1, OpenAI GPT-4o, Gemini 1.5, and Mixtral 8x7B are. These LLMs understand complex legal documents and further provide comprehensible answers to a wide range of legal queries on legal documents and policies. By integrating these powerful LLMs, the chatbot aims to bridge the gap between legal complexity and public understanding, offering users

interpretable legal information efficiently. Because the chatbot is designed to not store any sensitive information that users share, a legal assistant can help anyone at any time without worrying about data privacy.

Built user interactive User Interface using Streamlit. The Chatbot built using the streamlit framework as shown in figure 19 provides a chat box for users to input questions, the option to choose the model they prefer, and their experience with legal documentation so the answer can be fine-tuned as per the user's expertise with legal documents. A unique feature of the chatbot is the document upload functionality, which allows the user to upload a document and ask any question related to the document submitted. The conversation is kept for the individuals and as legal documents are sensitive information, the data is not stored for confidentiality.

Constraints

Data Collection. Fine-tuning the LLMs requires a huge amount of data to train the model for the use case. This process is time-consuming and challenging.

Fine-Tuning. With the recent research on the LLMs, a developer needs to be proficient in implementing fine-tuning.

Computational Resources. Training and deploying fine-tuned LLMS with a custom chatbot like LexLLM requires expensive access to powerful equipment, such as GPUs or TPUs.

Deployment. Integrating both RAG and the Fine-tuned model with the user interface has been significantly challenging.

Cost. Both software and hardware investments are substantial for the development and deployment of LexLLM.

User Experience. Gaining our users' acceptance requires developing an easy-to-use interface and providing comprehensive instructions on utilizing the LexLLM.

Validation. The LexLLM's performance, accuracy, and user satisfaction must be assessed to improve and optimize its replies.

System Quality Evaluation of Model Functions and Performance

In this section, LexLLM’s correctness and performance are evaluated, focusing on two key areas: the model's accuracy of its functions and the run-time performance in achieving system reaction time targets.

Correctness Evaluation

Correctness assessment is essential to ensuring that LexLLM consistently provides contextually relevant responses. To evaluate the correctness of LexLLM, a systematic approach was used to guarantee the accuracy and relevance of responses to the queries asked by the user. Human evaluations were used, which confirmed the accuracy of the model outputs by determining whether the responses were clear and acceptable for the context. By ensuring that LexLLM could provide accurate and contextually relevant responses, this evaluation method solidifies LexLLM's position as a successful legal assistant in intricate legal fields. Part of the correctness method also includes how well the model understands and generates relevant responses to user queries. GPT-4o, LLaMa 3.1, Gemini 1.5, and Mixtral 8x7b models were evaluated using important metrics such as perplexity, which gauges prediction confidence and lower values indicate better prediction certainty and OpenAI Eval where higher numbers indicate better accuracy.

Performance Evaluation

Response Time Distribution and Response Time Evaluation are two essential components in performance evaluation. These elements are crucial in ensuring LexLLM meets user needs for a seamless and effective conversation experience.

Response Time Evaluation

The focus of this evaluation is to examine how quickly the system could generate a response from the time the user submits a query in order to assess LexLLM's run-time performance to meet response time targets. Both the query's complexity and the model configurations, RAG or fine-tuned version had a significant impact on response times. LexLLM's infrastructure, which was optimized with T4 GPUs on Google Colab and AstraDB for speedy retrieval, was able to produce answers for simple queries in an average of 1-2 seconds. The RAG system, which extracts content chunks from lengthy legal texts, resulted in response speeds averaging 3 to 5 seconds for more complicated, multi-turn queries that needed deep legal analysis or multi-document retrieval. These response time targets are essential for ensuring responsive and user-friendly experiences.

System Visualization

The visualizations discussed below provide a structured view of LexLLM’s foundational data, the analysis conducted to refine this data, and the performance metrics critical to model selection and optimization.

Project Data

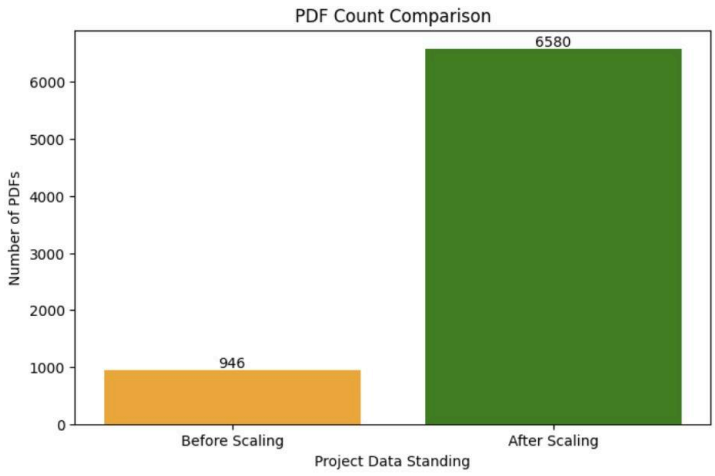
Comparative PDF Document Count. The scaling up of the data set that is used to train LexLLM's models is depicted as a bar plot in Figure 69. The dataset, which started out with 946

legal documents, is substantially increased to 6,580 documents now with the objective to improve the model's training and coverage of a broad range of legal aspects related to the tech

The significant spike in the volume of data is required to enhance the model's accuracy and responsiveness when dealing with diverse legal questions. By illustrating the scaling of data graphically, the plot offers an elementary, comprehensible representation of the project's data preparation progress. The model's ability to precisely comprehend complicated legal language was improved by the larger dataset size, which exposed it to a wider variety of legal terminologies, phrases, situations, and document structures.

Figure 69

PDF Data Counts



Note. Initial and Current Data Count. Original from Team 1.

Analysis Results

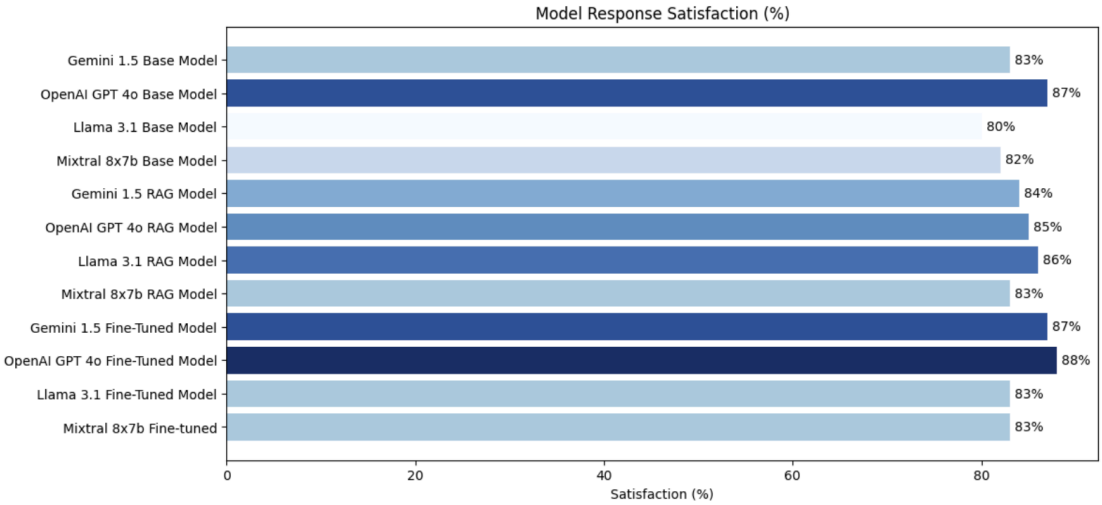
Model Performance Satisfaction. The analytical results of model performance in terms of satisfaction for the responses across models are displayed in Figure 70. The ability of each model to interpret and appropriately respond back to user queries in a tech oriented legal environment was the ultimatum of this evaluation.A survey was administered with a couple of

real-world legal questions, from simple definitions to intricate scenario-based queries, in order to assess the quality of the responses. A thorough set of feedback metrics are provided by the participants to measure user satisfaction by rating each model's response according to accuracy, relevance and clarity.

The results are compared to choose the model that provided the most contextually appropriate responses, which is represented by satisfaction metrics for each model. According to the research, the OpenAI GPT 4o Fine-Tuned Model had the highest satisfaction ratings, performing particularly well in legal-specific queries where preciseness and not apparent knowledge is essential. These insights are essential to identify the optimal configuration for deployment, guaranteeing the system's capability to accurately and consistently handle complicated legal concepts.

Figure 70

Satisfaction Rates



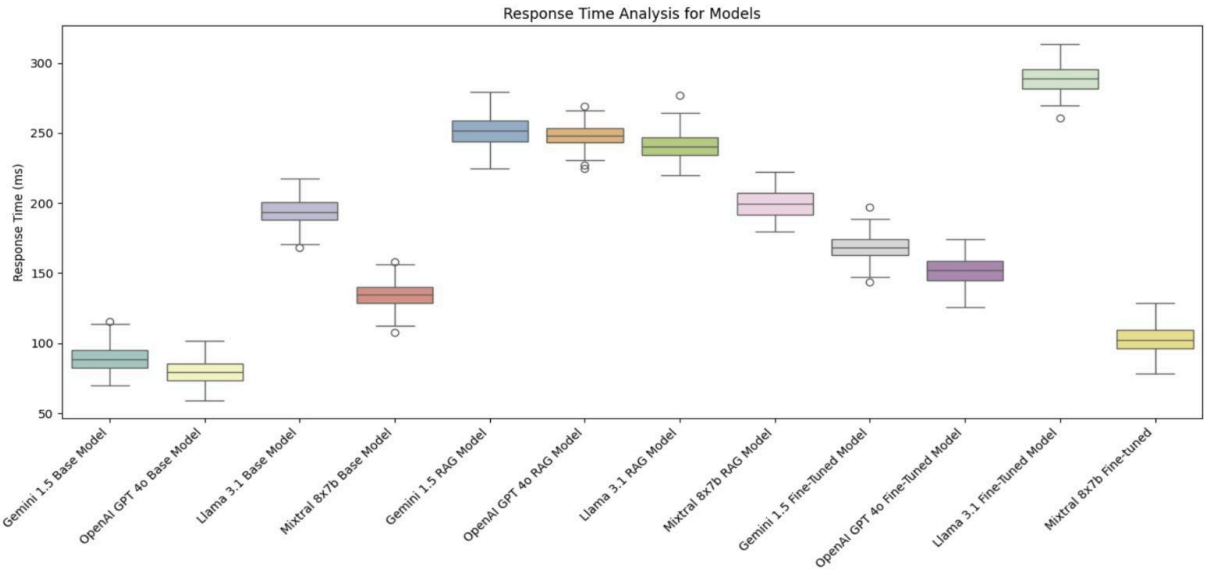
Note. Analytical Results of Model Performance for Satisfaction. Original from Team 1.

Machine Learning Outcomes

Response Time Comparison Across Models. The primary goal is to create a reliable and efficient legal document assistant that can consistently perceive intricate legal jargon and provide timely responses to user inquiries. Figure 71 compares the response times for multiple models under consideration, as it is vital to this effort. The computational efficiency of the model configurations, such as OpenAI, Llama, Gemini, and Mixtral, are clearly compared using the box plots. Llama and Gemini models consistently demonstrate superior processing rates, averaging between 150 and 200 milliseconds, whereas the visualization shows notable differences in response times.On the other hand, response times for the models OpenAI and Mixtral are considerably greater, typically over 250 milliseconds. As the agenda is to look for an optimal solution that strikes an acceptable balance between legal accuracy and performance efficiency in an effort to produce a high-performing legal text processing system, this comparative data is crucial to the final model selection process.

Figure 71

Response Time Analysis



Note. Comparing Response Time Across Models. Original from Team 1.

Conclusion

Summary

The project successfully tackled the issues that are faced by everyday citizens to keep themselves updated about the regular amendments by understanding the complexity associated with the legal frameworks and providing efficient solutions to easily navigate through the legislative codes, policies, and most of the historical cases. Through this solution citizens and even the lawyers can clarify their doubts or improve their knowledge about any legal information that they are either unfamiliar with or have limited knowledge of. By leveraging the RAGs and Fine-Tuned LLMs, the project implemented a chat application where users can either provide queries directly and receive the response from its knowledge base, or user can upload a PDF document containing legal information and LLMs can respond to the queries asked based on this

document. Some of the key tasks involved in achieving a properly working application are web scraping of data, performing extensive data cleaning, storing the properly processed data in the vector database, and choosing appropriate models for both RAG and fine-tuning ensuring that the final goal can be achieved successfully. The selection of the dataset was one of the most important steps as it helped to make models give more legal-focused responses to the queries. The user-friendly interface helped to keep users engaged and ensured accountability and transparency so that citizens could make informed decisions.

The LexLLM application holds the capability to improve the compliance of citizens with current laws and policies by allowing faster and more efficient methods to access legal documents which will also reduce the dependency on legal professionals to provide a response for the basic queries. It can aid researchers in accessing useful information for their research within a short period and can help policymakers access the existing legal laws and current policies followed by different companies to make better policies for their company which ensures the user's data is more secure and can also reduce the chances of any legal case filed by any user in the future.

Benefits and Shortcomings

The project offers numerous benefits, including easier access to legal documents such as laws, cases, and policies, which improves transparency for citizens and helps researchers work in the right direction. It also aids the citizens to ensure that they are compliant with all the laws, and provides lawyers easier access to past cases so that they can get a complete understanding of how the proceedings happened in that case based on different actions and then can decide what action to take in the current case which is very similar to that. By leveraging the concept of RAG and fine-tuning the performance of the model improved and it helped the model to become much

laser-focused on legal information. The user-friendly web interface helped to ensure that the users didn't get confused and could get responses to their queries very easily. The additional feature of selecting the experience level ensured that the user's level of legal understanding was kept in mind while generating the response.

Some limitations exist, such as the struggle that the models may have when responding to user queries that are related to ambiguous scenarios that have either been seen less or have never seen before, in these cases legal experts can provide better solutions or directions; need of a large number of computational resources to perform different tasks of projects like model fine-tuning, etc, which is addressed by scaling up the current resources; and the sense of insecurity or hesitation that user might have when uploading the PDF document having some sensitive information which is addressed in the way that the processed PDFs are stored in the persistent collection in vector database which is a temporary collection which gets refreshed as soon as the current user session ends. The response time by the system is greatly dependent on the internet speed which can affect the decision-making capability of the user and can also test their patience.

Potential System and Model Applications

The most important tasks in the legislative and legal sectors are maintaining a high volume of technical, text-heavy documents and observing complex legislation. System architectures and machine learning models can help in bringing efficiency, accuracy, and accessibility to this field in giant strides.

The LexLLM chatbots support users in understanding laws and legal policies while performing automated document processing to parse and summarize legal texts. While NLP-related research methods such as text summarization, and named entity identification review research and contracts, predictive analytics models predict outcomes for cases. These ensure that the sector is

developing into an accessible and efficient system characterized by better decision-making, cost reduction, and democratization of access.

Experience and Lessons Learned

The process of developing AI-powered application for legal and legislative domains has been challenging yet revolutionary and has provided insightful knowledge on user queries on text-heavy Datasets. The important learning during this project was about handling unstructured data and preprocessing and then structuring the data to ensure that the models generate accurate and contextually relevant answers. To successfully build the system the need for strong backends and an effective vector database was a major highlight and was integrated by RAG implementation. Preparing data in a specific format that each model requires for additional training was challenging while fine-tuning several models. This experience has confirmed the necessity of continuous learning and iterative development to effectively handle evolving needs.

Recommendations for Future Work

Efficiency Optimization

Research methods to improve the model's resource efficiency so that it can be applied when resources are constrained. Quantization, model distillation, and other techniques can reduce the model's size and processing needs without sacrificing performance.

Incorporating Diversity in Training Data

` Expanding the diversity of training data with a broad spectrum of different topics from different domains. By doing this, the model's ability to generate insightful and well-supported responses across a larger variety of inputs may be enhanced.

Improving Precision with Advanced Evaluation Metrics

Expand and further develop the set of evaluation metrics used for assessing the model's performance. Improving the scope of evaluation makes it more straightforward to assess the model's performance and guarantees not only technical accuracy but also the level of quality and ease of use of its output.

Advanced Multi-PDF Question and Answer Integration

Including the functionality of allowing for simultaneous analysis of multiple PDFs will increase cross-referencing and provide a streamlined workflow for research and analysis.

Dynamic Conversation Context Retention

Utilize dynamic context management methods to improve the model's ability to remember context over prolonged conversations. This could mean maintaining conversation histories more adeptly and keeping track of relevant information with greater efficiency.

Elevating Engagement with Multi-Modal Ability

Explore multi-modal features, such as the capacity to analyze and generate text alongside other forms of media like audio or images. A variety of new uses could result from this.

Integrating Legal Compliance and Updates

In the adaptive legal landscape, integrating APIs or databases can ensure the LexLLm application always reflects current laws and regulations. In addition to reducing the possibility of inaccurate or out-of-date information, this dynamic refreshing mechanism makes the system a reliable resource for both ordinary users and legal specialists.

Contributions and Impacts on Society

Cultural Contributions

Democratizing Legal Access. Lowers the barriers to understanding and utilizing legal information. One of the functionalities of the application is to capture the awareness of the user's familiarity with the legal document making responses cater to the audience of the application.

Empowering Legal Awareness. By making difficult legal ideas understandable to the general public, LexLLM promotes an understanding of legal literacy. This promotes educated civic engagement and helps create a society where people are more aware of their responsibilities and rights.

Enhancing Ethical Practices. By making legal and regulatory compliance easier, LexLLM promotes ethical behavior in both individuals and organizations. This encourages a culture of integrity and accountability.

Educational Contributions

Enhancing Legal Education. LexLLM becomes a useful educational resource by providing concise, understandable explanations of laws, precedents, and legal ideas. It can promote increased legal literacy by helping the public, educators, and students comprehend difficult legal issues. People are better able to interact with the legal system in a society that is better informed.

Assistance with Writing and Research. Assist students and researchers in developing concepts, refining their writing, and discovering resources to raise the standard and effectiveness of their research.

Economic Contributions

Lowering Legal Costs. LexLLM makes legal services more accessible to individuals and small enterprises by reducing the need for costly legal consultations for everyday situations. This promotes financial resiliency and economic empowerment.

Promoting Economic Participation. With LexLLM's assistance, small and medium-sized businesses, students, and legal professionals can more successfully manage regulatory environments.

Fostering Social Well-being

Accessibility for People with Disabilities. People with disabilities can use and benefit from LexLLM's services because of its input possibilities, and user-friendly design. The judicial system is more responsive to the needs of all citizens and more open as a result of this inclusivity.

Empowering Volunteer Legal Services. The capacity of volunteer attorneys and non-profit organizations to assist their clients can be significantly enhanced by LexLLM. It helps lawyers to handle more cases more effectively by simplifying preliminary research and providing accurate, context-relevant guidance.

Promoting Transparency and Accountability

Increased transparency in court procedures is facilitated by the application's capacity to access and evaluate enormous volumes of legal data. It can assist in spotting trends in case results, spotting possible prejudices, and bolstering initiatives to make legal systems more equitable and consistent.

Impacts on Culture and the World

In regions with limited access to legal resources, these applications can serve as an alternative to traditional legal support, providing underserved populations with vital tools to navigate their local legal systems. This improves justice and equity on a global scale.

References

Adams, L., Busch, F., Han, T., Excoffier, J., Ortala, M., Löser, A., Aerts, H. J., Kather, J. N., Truhn, D., & Bressem, K. (2024). LongHealth: A Question Answering Benchmark with Long Clinical Documents. *arXiv.org*. https://doi.org/10.48550/arXiv.2401.14490

Ainslie, J., Lee-Thorp, J., de Jong, M., Zemlyanskiy, Y., Lebron, F., & Sanghai, S. (2023). GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. *In the Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. https://doi.org/10.48550/arXiv.2005.14165.

Alan, A. Y., Karaarslan, E., & Aydin, Ö. (2024). A RAG-based question answering System Proposal for Understanding Islam: Mufassir QAS LLM. *arXiv.org*. https://doi.org/10.48550/arXiv.2401.15378

Bagshaw, K. B. (2021). NEW PERT and CPM in Project Management with Practical Examples. *American Journal of Operations Research, 11*(04), 215–226. https://doi.org/10.4236/ajor.2021.114013

Bhattacharya, P., Hiware, K., Rajgaria, S., Pochhi, N., Ghosh, K., & Ghosh, S. (2019). A comparative study of summarization algorithms applied to legal case judgments. *Lecture Notes in Computer Science*,413–428. https://doi.org/10.1007/978-3-030-15712-8_27

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., . . . Amodei, D. (2020).

Language Models are Few-Shot Learners. *In the Proceedings of the 2020 Conference on Neural Information Processing System, 159,* 1877-1901. https://doi.org/10.48550/arXiv.2005.14165

Chen, A., Yao, F., Zhao, X., Zhang, Y., Sun, C., Liu, Y., & Shen, W. (2023). EQUALS: A real-world dataset for legal question answering via reading chinese laws. *In Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law ,* 71-80.

Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., …Le, Q.V. (2022) Scaling Instruction-Finetuned Language Models. *arXiv.org*. https://doi.org/10.48550/arXiv.2210.11416

Dao, T., Y. Fu, D., Ermon, S., Rudra, A., & Re, C. (2022). FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. *In the Proceedings of the 2022 Conference on Neural Information Processing System.* https://doi.org/10.48550/arXiv.2205.14135

Gallegos, I., & George, K. (n.d.). The right to remain plain: summarization and simplification of legal documents. *In Stanford University*, *Stanford CS224N Custom Project*. https://web.stanford.edu/class/cs224n/reports/custom_116652906.pdf

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., & Leahy, C. (2020). The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv.* https://doi.org/10.48550/arXiv.2101.00027

Gemini Team. (2023). Gemini: A Family of Highly Capable Multimodal Models. https://doi.org/10.48550/arXiv.2312.11805

Gemini Team. (2024) Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. https://doi.org/10.48550/arXiv.2403.05530

Jain, S., Madeleine, V. Z., Hajishirzi, H., & Beltagy, I. (2020). SCIREX: A Challenge Dataset for Document-Level Information Extraction. *arXiv.org*. https://doi.org/10.48550/arXiv.2005.00512

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Renard Lavaud, L., Lachaux, M.-A., Stock, P., Le Scao, T., Lavril, T., Wang, T., Lacroix, T., & El Sayed, W. (2023). Mistral 7B. https://doi.org/10.48550/arXiv.2310.06825

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Bou Hanna, E., Bressand, F., Lengyel, G., Bour, G., Lample, G., Renard Lavaud, L., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Le Scao, T., Gervet, T., Lavril, T., Wang, T., Lacroix, T., & El Sayed, W. (2024). Mixtral of Experts. https://doi.org/10.48550/arXiv.2401.04088

Llama Team (2024). The Llama 3 Herd of Models. https://doi.org/10.48550/arXiv.2407.21783

Louis, A., Van Dijck, G., & Spanakis, G. (2023). Interpretable Long-Form Legal Question Answering with Retrieval-Augmented Large Language Models. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2309.17050

Medeiros, T., Medeiros, M., Azevedo, M., Da Silva, M. B. D., Silva, I., & Costa, D. G. (2023). Analysis of Language-Model-Powered chatbots for query resolution in PDF-Based automotive manuals. *Vehicles*, *5*(4), 1384–1399. https://doi.org/10.3390/vehicles5040076

Miller, D. M. (2019). Leveraging BERT for extractive text summarization on lectures. *arXiv.* https://doi.org/10.48550/arXiv.1906.04165

Nay, J. J., Karamardian, D., Lawsky, S. B., Tao, W., Bhat, M. M. A., Jain, R., Lee, A. T., Choi, J. H., & Kasai, J. (2023b). Large Language Models as Tax Attorneys: A Case Study in Legal Capabilities Emergence. *Social Science Research Network*. https://doi.org/10.2139/ssrn.4476325

OpenAI Team (2023). GPT-4 Technical Report. https://doi.org/10.48550/arXiv.2303.08774

Pandya, K., Holia, M., (2023). Automating Customer Service using LangChain. *arXiv.org*. https://doi.org/10.48550/arXiv.2310.05421

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners.

Saad-Falcon, J., Barrow, J., Siu, A., Nenkova, A., Yoon, D. S., Rossi, R. A., & Dernoncourt, F. (2023). PDFTriage: Question Answering over Long, Structured Documents. *arXiv.org*. https://doi.org/10.48550/arXiv.2309.08872

Saito, K., Sohn, K., Lee, C.Y., Ushiku, Y. (2024).Unsupervised LLM Adaptation for Question Answering. *arXiv.org*. https://doi.org/10.48550/arXiv.2402.12170

Shazeer, N. (2020). GLU Variants Improve Transformer.

https://doi.org/10.48550/arXiv.2002.05202

Shim, Y., Philippides, A., Staras, K., & Husbands, P. (2016). Unsupervised Learning in an Ensemble of Spiking Neural Networks Mediated by ITDP. *PLOS Computational Biology, 12*(10). http://dx.doi.org/10.1371/journal.pcbi.1005137.

Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., & Liu, Y. (2021). RoFormer: Enhanced Transformer with Rotary Position Embedding. https://doi.org/10.48550/arXiv.2104.09864

Team, G. H. C., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., Millican, K., Silver, D., Petrov, S., Johnson, M., Antonoglou, I., Schrittwieser, J., Glaese, A., Chen, J., Pitler, E., . . . Pillai, T. S. (2023). Gemini: a family of highly capable multimodal models. *arXiv*. https://doi.org/10.48550/arxiv.2312.11805

Topsakal, O., & Akıncı, T. Ç. (2023). Creating large language model applications Utilizing LangChain: A primer on developing LLM apps fast. *International Conference on Applied Engineering and Natural Sciences, 1*(1), 1050–1056. https://doi.org/10.59287/icaens.1127

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, É., & Lample, G. (2023). LLAMA: Open and Efficient Foundation Language Models. https://doi.org/10.48550/arxiv.2302.13971

Touvron, H., Martin, L., Stone, K. H., Albert, P. J., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M.,

Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., . . . Scialom, T. (2023). Llama 2: Open foundation and Fine-Tuned chat models. https://doi.org/10.48550/arxiv.2307.09288

Turtle, H. R. (1995). Text retrieval in the legal world. *Artificial Intelligence and Law*, *3*(1-2), 5–54. https://doi.org/10.1007/bf00877694

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. (2017). Attention is All You Need. *Advances in Neural Information Processing Systems, 30.* https://doi.org/10.48550/arXiv.1706.03762

Wikipedia contributors. (2023, December 28). *Westlaw*. Wikipedia. https://en.wikipedia.org/wiki/Westlaw

Xiao, C., Hu, X., Liu, Z., Tu, C., & Sun, M. (2021). Lawformer: A pre-trained language model for Chinese legal long documents. *AI Open*, *2*, 79–84. https://doi.org/10.1016/j.aiopen.2021.06.003

Zhang, B., & Sennrich, R. (2019). Root mean square layer normalization. *In the Proceedings of the 2019 Conference on Neural Information Processing System.* https://doi.org/10.48550/arXiv.1910.07467

Appendix A

System Testing

The testing results below showcase the primary GUI components in the LexLLM application, with expected and actual outcomes for each use case.

Required Use Case: User Authentication

Test Case 1: Login using Google Outh

1. Navigate to the Login page at the main URL
2. Click on “Continue with Google”

Expected Result: The user is successfully logged in, and the confirmation message is displayed “Login successful!”. The user is redirected to the main screen of the application.

Actual Result: The user when presses on the button, using Descope platform the authorization is performed, where selection of Gmail ID is done by the user and when that is successfully completed the “loading” message is shown for few seconds and then the confirmation message is displayed and the user is redirected to main screen of the application.

Test Case 2: Login with Valid Credentials (Admin)

1. Navigate to the Login page at the main URL.
2. Enter a registered Username and Password.
3 Click the Login button.

Expected Result: The user is successfully logged in, and a confirmation message, “Login successful!” is displayed. The user is redirected to the main screen of the application.

Actual Result: The login was successful, and the confirmation message displayed as expected. The user was redirected to the main screen of the application without issues.

Test Case 3: Log in with Invalid Credentials (Admin)

1.Navigate to the Login page.
2.Enter an incorrect Username or Password.
3.Click the Login button.

Expected Result: Login fails with the error message "Invalid username or password."

Actual Result: The system displayed the error message "Invalid username or password," and the user was prevented from logging in as expected.

Test Case 4: Logout Functionality

1.From the main application screen, click the **Logout** button in the sidebar.

Expected Result: User is logged out, session state is cleared, and the user is redirected back to the Login page.

Actual Result: The logout function worked as expected. The session state cleared, and the user was redirected to the Login page.

Required Use Case: Chat Interaction with PDF Support

Test Case 1: Upload PDF and Display Context

1.Navigate to the sidebar.
2.Select and upload a PDF file.

Expected Result: PDF text is extracted, stored in the backend, and the success message, "PDF uploaded, processed, and stored successfully!" is displayed.
Actual Result: The PDF was uploaded and processed successfully. The success message appeared, confirming storage, and text was accessible for queries.

Test Case 2: Ask a Question Using PDF Content as Context

1.Type a question into the chat input that requires context from the uploaded PDF.
2.Press Send.

Expected Result: Response generated based on the PDF context.

Actual Result: The chatbot provided a relevant response based on the PDF content. The response was contextually accurate and used information from the uploaded file effectively.

Required Use Case: Model Selection and Chat History Persistence

Test Case 1: Select Model for Chat

1.In the sidebar, select "Llama 3.1 RAG" (for instance) from the model options.
2.Type a question in the chat and press Send.

Expected Result: The chatbot generates a response using Llama 3.1 RAG.

Actual Result: After selecting Llama 3.1 RAG, the chatbot generated a response, demonstrating that the correct model was in use.

Required Use Case: Experience Level Selection

Test Case 1: Select Experience Level for Chat

1. In the sidebar, select the experience level "Beginner" (for instance).

2. Type the question in the chat and press “Enter” key.

Expected Result: The chatbot generates the response by using less legal jargons since the experience level of the user selected as Beginner.

Actual Result: After selecting the experience level as Beginner, the chatbot will use the selected model from the dropdown present in the sidebar and generate a response, which is having less legal jargons and easier for the users like normal citizens to understand demonstrating that the correct experience level was used.

Appendix B

Project Data Source and Management Store

Table B1

Folders and their Links

Folder	Links
Raw Data	https://drive.google.com/drive/folders/1aGi9cmuCG91Q6RgRralr3Ve7w9q2vuP7?usp=drive_link
QA	https://drive.google.com/drive/folders/1YDD52IBfC3ayU-HcqQbDLpAL3daTFP_X?usp=drive_link

Note. The table contains the folder and their corresponding links. Original from Team 1.

Appendix C

Project Program Source Library, Presentation, and Demonstration

Table C1

Deliverables and their URLs

	URL
Project Source Library	https://drive.google.com/drive/folders/0AKisc7vcbrxzUk9PVA?dmr=1&ec=wgc-drive-globalnav-goto
Source Code	https://drive.google.com/drive/folders/1rPWZ10yFtgGhOf4jIbSYyDAZDEIAYp_Q?usp=drive_link
Presentation	https://drive.google.com/drive/folders/1Y8B-u3FFNOqWmBhLC5JQk6p1RDLlxs_x?usp=drive_link
Demo (Videos)	https://drive.google.com/drive/folders/1w5H6xw3y2x8Cu-tqTk6r13BzO15rbRiw?usp=drive_link

Note. The table shows different deliverables and their URLs. Original from Team 1.