LawTalk : A Portuguese Lawyer

Afonso Bessa, Francisco Claudino, Mariana Marques, Eduardo Henriques

Abstract

The development of LawTalk, an advanced legal chatbot designed specifically for Portuguese law, represents a significant leap in the integration of Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG). This innovative approach overcomes the typical limitations of LLMs, such as outdated information and hallucinations, by dynamically incorporating pertinent external data to produce precise and contextually accurate responses. LawTalk is engineered to improve access to legal information, delivering timely and accurate legal advice while maintaining transparency through the citation of relevant legal articles. The project focuses on creating an intuitive user interface and optimizing the chatbot for a wide range of legal contexts. Preliminary evaluations reveal that Llama2, among the tested models, excels in accuracy, response time, and consistency. Future enhancements will include expanding the chatbot's knowledge base, refining the user interface, and enabling the continuity of user interactions to further personalize and enhance the user experience.

Keywords: ChatBot, LangChain, Large Language Models, Ollama, Portuguese Law, Retrieval Augmented Generation

1 Introduction

The evolution of Natural Language Processing (NLP) and Machine Learning (ML) technologies has opened new possibilities for creating more intelligent and useful virtual assistants. In this context, Large Language Models (LLMs) like Llama2, developed by Ollama [1], or GPT 4.0 by OpenAI [2], have proven to be particularly effective in understanding and generating text in a very natural and accurate way. At the same time, Retrieval-Augmented Generation (RAG) has emerged as a promising technique to enhance the performance of these models by integrating external and domain-specific information.

The integration of RAG further enhances chatbots by incorporating external information sources, thereby improving the quality of responses and providing up-to-date and contextually relevant information. RAG combines retrieval models with Large Language Models (LLMs) to address the limitations of LLMs, such as hallucinations and

outdated knowledge, by dynamically retrieving and incorporating relevant external information into the generation process. This approach results in significant improvements in knowledge-intensive tasks and various applications, including question answering and conversational systems, ensuring that chatbot responses are accurate and grounded in the latest information [3].

In the legal domain, chatbots offer both significant benefits and notable limitations. They enhance accessibility to legal information, aiding those without professional counsel, and can expedite data retrieval while navigating users through intricate legal processes. However, the critical importance of accuracy to prevent possible legal consequences, combined with a lack of empathy, requires a careful balance between providing information and delivering formal advice. Past negative experiences with chatbots may also lead to trust issues, casting doubt on their reliability in delivering comprehensive legal assistance. Despite these challenges, a well-designed legal chatbot can revolutionize access to justice by providing timely and accurate information, assisting in solving practical cases, and citing specific legal articles [4].

Given these considerations, the primary objectives of this project are to develop a legal chatbot capable of solving practical portuguese legal cases and providing information about the legal consequences for offenders. The chatbot will be designed to cite the relevant legal articles it relied on to resolve these practical cases, ensuring transparency and reliability. Additionally, the project aims to create a practical chatbot that responds in natural language, thereby enhancing the user experience. Furthermore, exploring customization options for LLMs and embeddings will be essential to optimize the chatbot for different legal contexts, ensuring it meets the specific needs of various users and jurisdictions.

Regarding the remaining structure of this document, the following section explores the latest developments in legal chatbots. Then, in the third section, describes the architecture and key components of our legal chatbot system, detailing the methodologies and development tecnologies. The fourth section discusses the data sources and the processes involved in processing the data. The fifth section presents the results of our analysis, highlighting the impacts and effectiveness of the legal chatbot. Finally, in the sixth section the results are analysed, the conclusions obtained are presented, and future works reletaded to this study are suggested.

2 State of the Art

In this section, a review of existing and recent projects related to legal systems will be made.

2.1 Quick Overview of Recent Developments in Legal Chatbots

In recent years, several chatbots powered by advanced technologies have been developed to enhance access to legal services, streamline legal processes, and improve overall efficiency in the legal industry.

DoNotPay is one of the most well-known legal chatbots. Initially launched in the United States to help users contest parking tickets, it has expanded to cover a wide range of legal issues including subscription cancellations, small claims court filings, and more. The bot guides users through legal procedures by asking relevant questions and

generating necessary documents. This tool democratizes legal assistance by making it more affordable and accessible, especially for those who cannot afford traditional legal services [3, 5].

LawDroid is another significant player in the legal chatbot space, primarily used in the United States and Canada. It helps law firms automate client intake, appointment scheduling, and document drafting. LawDroid can quickly guide clients through complex legal queries, providing step-by-step assistance and legal information. This chatbot is particularly useful for law firms looking to increase efficiency and focus more on high-value tasks by automating routine interactions and paperwork [4].

Alpaca Law is a legal chatbot specifically designed to assist with legal research and document preparation in Portugal. Launched recently, it has quickly gained popularity with over 530 users and more than 1,700 chats within its first two weeks. Alpaca Law uses natural language processing to interpret legal questions and provide answers based on Portuguese law. It accesses all legislation published in the *Diário da República* [6], offering users relevant legal information and the specific laws consulted. This tool is particularly useful for the average user needing legal answers without registration requirements, as well as legal professionals and academics who need to navigate and summarize complex legal texts efficiently. However, while Alpaca Law is a valuable research aid, it cannot replace professional legal advice and should be used as a supplementary tool for initial legal inquiries.

3 Methods and Development

This section provides an overview of the methodology and development processes used in our solution, including system architecture, LLMs, and software tools.

3.1 Methodology

The RAG methodology used in this project allows LawTalk to access a vast amount of specific data on portuguese law, using this information to generate more accurate and relevant responses. On the other hand, the use of LLMs enabled natural language interaction with users, ensuring an intuitive and efficient user experience. To ensure all of that the following steps were made:

1. **Research and Data Selection**: Identification and organization of reliable data sources on portuguese law.
2. **Development of the RAG-LLM**: Configuration and training of the model by integrating the capabilities of RAG with natural language generation of LLM.
3. **Chatbot Development**: Implementation of the chatbot using the pre-trained model, focusing on providing precise and personalized responses.
4. **Testing and Validation**: Evaluation of the chatbot's performance through practical tests and user feedback.
5. **Graphical User Interface Development**: Implementation of a User Interface that allows the user to interact whit the chatbot in a Web Based Application.

3.2 Architecture

As shown in Figure 1, the architecture of the system comprises three main components: the **Chatbot Interface**, implemented with the Streamlit library; the **Database**, which is ChromaDB; and the **LangChain Framework**, which utilizes the Ollama LLMs. Each of these components is crucial to the system's structure and serves distinct functions, as detailed below:

- **ChatBot Interface:** Interacts with the user as well as with ChromaDB and LangChain and is responsible for receiving user questions, send the user inputs to the database and deliver the answer to the User.
- **ChromaDB:** Stores and manages all the data needed for the ChatBot, including PDFs, questions and answers that were previously generated by the LLMs, storing them in the form of vector embeddings. It also acts as an intermediary between LawTalk and LangChain, facilitating data storage and retrieval.
- **LangChain:** A language model framework that processes user input, extended context generated by RAG, and memory to generate accurate answers. Utilizes advanced language processing capabilities to understand and respond to user inputs.
- **Ollama LLM:** These models are used by LangChain to generate the answer to the user's question, by feeding them with the extra content selected by the retriever in the retrieval process, the memory which comes from the previous generated answers and the user question.

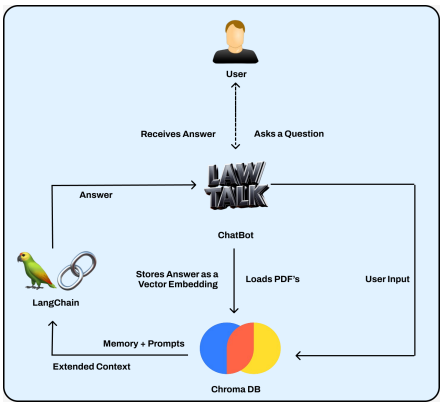

Fig. 1 System Architecture

The diagram presented in the Figure 2 helps to better understand the system functionality. When the LawTalk chat starts, it loads every document in the folder *Final PDF Files* into ChromaDB to be stored as vector embeddings (1 in the Figure 2). The Portable Document Format (PDF) files and the methods utilized for their selection and data processing will be explained in the Section 4. Following this, the user asks a question to the LawTalk. The chatbot reads the user input and stores it in the vector database, as well as the LLM chosen to answer the question (2 and 2.1 in the Figure 2). Then, the retriever re-ranks the vectors that contain the information from the documents to

extract extra content based on the user input (this method is the retrieval that is part of the RAG Framework). Then, the database gives LangChain the user input, the extra content, the memory (which contains the answers that were previously generated by the LLM), the LLM and the answer prompt which is utilised by this framework to generate the answer (2.1.1, 2.1.2 and 2.1.3 in the Figure 2). Then the answer is provided to the user and it is stored in the database, as a vector embedding (2.1.3.2 in Figure 2).

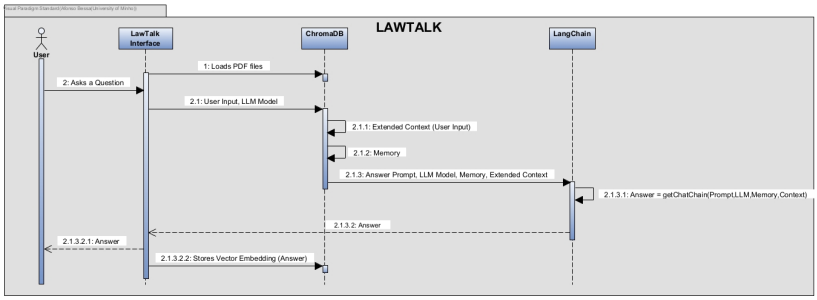

Fig. 2 Flow Diagram

3.3 Technologies

The tools used in this project were not chosen at random; there are several reasons why they were selected, as follows:

- **ChromaDB** is a database designed for easy and manageable access to vector embeddings, chosen as the primary way to store the data fed to the LLMs for several reasons. These include efficient representation by reducing high-dimensional, sparse representations (e.g., one-hot encoding) to dense, lower-dimensional vectors, making computations more efficient and reducing memory requirements. Under the hood, it uses the SQLite library which allows for efficient file storage with minimal metadata, a key attribute when manipulating large ammounts of external data. ChromaDB captures semantic meaning and syntactic properties in a compact form, making it easier to process and understand relationships between words. It also enables models to understand and group similar words or concepts based on their contexts, improving tasks like clustering, classification, and recommendation. Additionally, ChromaDB can be adapted to specific domains or languages with relatively little additional training, enhancing its applicability across different fields [7].
- **LangChain** offers a range of advantages when training and deploying LLMs, making it a valuable tool for developers working on advanced NLP applications. It provides modular components for various stages of LLM development, such as data preprocessing, model training, and evaluation, allowing for easier experimentation and customization. LangChain is designed to be intuitive and easy to use, reducing the learning curve for developers new to LLMs. It offers optimized workflows for common tasks such as data loading, preprocessing, and model training, leading to significant time savings and improved performance. Furthermore, LangChain ensures that models can be trained and deployed on large datasets efficiently [8].

- **Ollama LLMs** are open-source and offer several advantages for developing a chatbot, making them a compelling choice for creating sophisticated, efficient, and effective conversational agents. These models maintain and utilize context over multiple turns in a conversation, leading to more coherent and contextually relevant responses, and this trait can be expanded upon to store information about multiple full past conversations into the model's memory. They allow for personalization to adapt the chatbot's tone, style, and responses to match user preferences, creating a more engaging user experience. Additionally, they leverage advanced natural language generation capabilities to produce responses that are more natural and human-like, improving user satisfaction [1].
- **Streamlit** is a powerful Python library that enables developers to quickly and effortlessly create interactive and visually appealing Graphical User Interfaces (GUI). This tool is invaluable for creating front-end applications that can interact with machine learning models and data analytics workflows, providing an intuitive interface for end-users [9].

3.4 LLMs

In LawTalk, we provide the user with the option to use one of three LLMs provided by Ollama: **Mistral**, **Llama2**, and **Zephyr**.

Mistral is a robust and highly versatile language model designed to handle a wide range of NLP tasks. Known for its balanced performance across various applications, Mistral excels in generating coherent and contextually relevant text. It is well-suited for both creative and analytical tasks, making it a popular choice for users seeking a dependable and flexible language model that can adapt to diverse linguistic challenges.

Llama2 is the second iteration of the Llama language model series, offering enhanced capabilities and improved performance over its predecessor. It is particularly noted for its ability to understand and generate complex and nuanced text, making it ideal for sophisticated tasks that require deep comprehension and detailed responses. Llama2's advanced architecture enables it to perform exceptionally well in scenarios demanding high precision and contextual awareness.

Zephyr stands out as a cutting-edge language model known for its speed and efficiency in processing and generating text. It is optimized for real-time applications where rapid response times are crucial. Zephyr's streamlined design ensures that it can handle high-volume requests with minimal latency, making it a preferred option for applications requiring quick and reliable language processing, such as chatbots and live support systems.

The performance and quality of answers will be discussed in Section 5.

4 Data Sources

In the development of this sophisticated chatbot, one of the most critical aspects was the selection and utilization of documents to feed the LLM. These documents form the foundational knowledge base from which the chatbot derives its understanding and generates responses. High-quality, relevant and comprehensive documents ensured that the LLM was well-informed, enabling the chatbot to provide accurate, contextually relevant and reliable answers to user queries.

4.1 Data Selection

The field of criminal law is composed of substantive criminal law, which includes matters related to types of crimes, legal consequences of crimes, etc., and procedural criminal law, which deals with how criminal proceedings are conducted in Portuguese courts. For this chatbot, we used theoretical notes from classes at the Faculty of Law of the University of Porto on the field of criminal law, both substantive and procedural, as well as Portuguese legislation on criminal law, including the Penal Code and the Code of Criminal Procedure. The former contains the different types of crimes and their respective consequences, while the latter deals with the conduct of criminal proceedings from the moment the investigation is initiated by the Public Prosecutor's Office to the appeal of the decisions rendered, among other issues. Additionally, we used practical cases and their respective answers to enhance the chatbot with more concise and accurate information.

4.2 Data Processing

After selecting the articles we wanted to feed to the LLM, it was necessary to extract the information and format it in a specific way for the model to learn more effectively and efficiently.

Since the large files were not being well understood by the model, it was necessary to split them into smaller chunks to improve processing. To achieve this, a Python script was developed to automatically divide the files according to their corresponding sections and apply a deep cleaning process to remove noise by eliminating headers, footers, and page numbers. Subsequently, we performed character normalization, converting all text to lowercase using regular expressions.

Afterwards, in order to process the content of each chunk, we utilized three Python libraries to extract information from PDF files to text files. These libraries were:

- **PyPDF2** - allows the manipulation of PDF files. It is used for tasks such as extracting information like text and metadata and splitting documents page by page [10].
- **PyTesseract** - is an open-source Optical Character Recognition (OCR) engine that can recognize and extract text from images, scanned documents, and PDF files. It is one of the most accurate and widely used OCR solutions available for image scanning [11].
- **PDFPlumber** - is designed for extracting information from PDF files with high precision. It is especially useful for dealing with complex PDF layouts, such as those containing tables, forms, and embedded images. Unlike some other PDF extraction tools, PDFPlumber is known for its ability to handle the extraction of structured data in a more granular and accurate manner [12].

Next, we developed a Python script for each of the aforementioned tools and conducted extensive testing. We determined that PDFPlumber was the optimal choice for our needs due to its superior ability to handle complex PDF layouts with high precision, resulting in fewer spelling errors. In comparison, PyPDF2 lacked the same level of accuracy and detail, while PyTesseract was more suited for image scanning. Given that our documents required precise table and form extraction rather than image extraction, PDFPlumber clearly emerged as the best tool for the job.

Following the extraction phase, we organized the files by title, chapter, section and article format. This meticulous formatting ensured that the legal texts were structured logically and were easy to process by the LLMs.

After formatting, the text files were processed and translated from Portuguese to English. This translation step was crucial as LLMs tend to learn better in English due to the vast availability of high-quality training data, the relatively straightforward linguistic structure of English, and significant resource allocation by major tech companies towards English-language models. The abundance of English content on the internet and its global dominance in schools, technology, and business further enhance the performance of these models in English compared to other languages. We considered procuring a model that was specifically built to learn Portuguese information in order to skip this step, but available models are scarce and lack the robustness seen in the more popular choices.

When translating the text, we analyzed various Python libraries that accomplish this task: `googletrans` [13], `translate` [14], `translators` [15] and `deep_translate` [16]. Of these four libraries, `deep_translate` offers a variety of translators of varying quality and efficiency, and has a very high monthly character limit of 500 000, so it was our final choice.

Finally, the translated and processed files were converted back into PDFs. This ensured that the final documents were both accessible and useful for future reference, maintaining the integrity of the information while leveraging the enhanced learning capabilities of the models.

The model retains the English input and answer in it's memory due to the reasons discussed above, the main one being the enhanced English learning capability. The output that is presented, however, is translated back to Portuguese to provide a better user experience. The previously chosen library works well with large amounts of text but responses are usually much shorter, so the lighter `translators` library was more adequate to use here.

5 Results and Discussion

At the end of the project, we implemented a GUI to deploy our project as a web-based application using the Streamlit library. A GUI is a common feature of most successful chatbots, as it allows for a more streamlined user experience.

As seen below, our interface allows for users to alter the model type between the three main models that we tested (*mistral*, *llama2*, *zephyr*) and allows for the embeddings to be dynamically reloaded, effectively erasing the chatbot's memory.

With the interface completed, the final stage of our project involves retrieving data from each model and comparing the results to determine which one is better in terms of categories like **answer quality and correctness**, **generation time**, and especially **consistency**. Balancing and maximizing these factors is crucial to ensuring the highest possible quality of interactions.

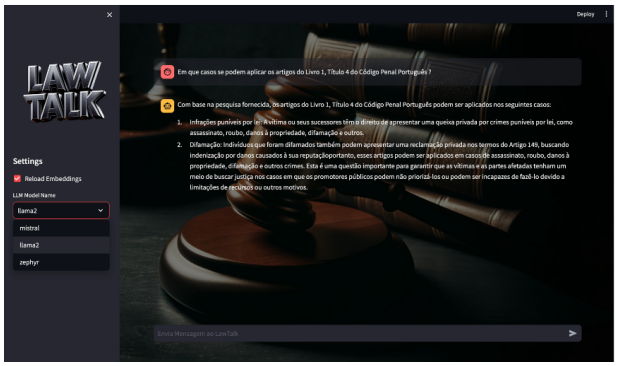

Fig. 3 Graphical User Interface

We created a dataset based on the processed files, consisting of 50 questions and answers in a Comma-Separated Values (CSV) file. Each model will answer the questions, and its answers will be compared to the real ones to obtain a performance rating from 1 to 10: 1 corresponding to a likely hallucination and 10 being a very similar response to the real answer. We will also measure the time taken to generate each output.

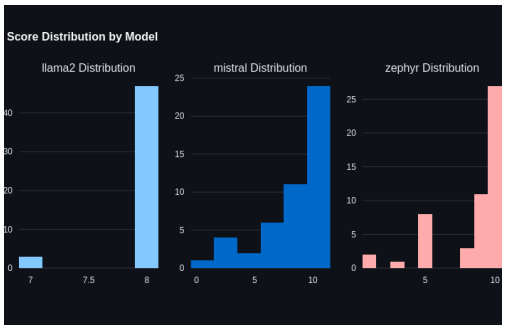

Fig. 4 Scores measured for each model

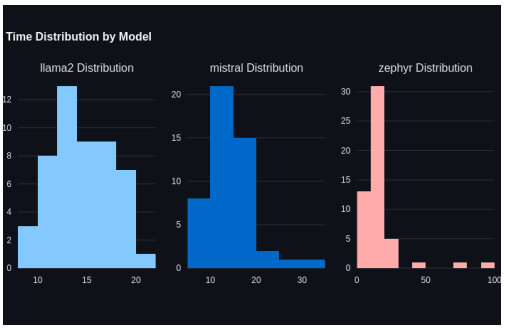

Fig. 5 Measured response times

By analyzing the results above, it is clear that llama2 is by far the most consistent model, with about 9 out of 10 answers receiving a score of 8. During testing without using our dataset, we managed to obtain at least one score of 9, indicating that most, if not all, outputs will have a score ranging from 7 to 9. The generation times were also extremely consistent, all being under or around 20 seconds. This places llama2 at the top of our ranking, making it more than capable of handling everyday tasks in this domain.

On the other hand, the two remaining models proved to be much less consistent, despite providing superior answers to about 50% of questions. These models produced both perfect answers and hallucinations when using the same dataset as llama2. However, between the two, mistral provides great answers more consistently.

They are also generally worse in terms of the time spent generating the output, as it can reach up to 30 seconds for mistral and up to 100 seconds for zephyr. Higher response

times also correlate to lower output quality in zephyr's case. Both models could not beat llama2's consistently low times, with some prompts taking consider to be fully processed. We suspect that zephyr is quite inconsistent because it is a fine-tuned version of mistral, designed for very specific tasks such as text summarizing, and always presents outside sources when generating its responses. Because of this, the model's reasoning capabilities are significantly hampered.

Despite the results shown above, automated output validation is inferior to manually confirming the correctness of the answers. Therefore, a perfect score does not explicitly imply a perfect answer. Human validation will always be more reliable in this regard, and when manually analyzing most answers, we can conclude that **llama2 is indeed the superior model** for this specific domain.

6 Conclusion and Future Improvements

In conclusion, the development of LawTalk, an advanced chatbot leveraging RAG and LLMs, has demonstrated significant potential in solving practical Portuguese legal cases and providing information about the legal consequences for offenders. By integrating cutting-edge technologies such as the Ollama LLMs, ChromaDB for vector storage, and LangChain for model management, we have created a robust system capable of delivering accurate, relevant, and timely responses to user questions. The chatbot cites the relevant legal articles it relies on to resolve cases, ensuring transparency and reliability. Its natural language responses enhance the user experience, and customization options for LLMs and embeddings optimize the chatbot for different legal contexts and jurisdictions as well as different hardware specifications. More capable machines would naturally want to run models with more parameters, as this usually translates into more accurate responses.

A crucial factor in the success of this project was the intensive effort in processing every file. This involved splitting large files into smaller chunks to facilitate efficient learning and processing by the LLMs, extracting and formatting the data to ensure optimal model performance, and translating the content to enhance comprehension.

Our extensive evaluation of the LLMs highlighted the strengths and areas for improvement in each model. The comparison between Mistral, Llama2, and Zephyr revealed that while Llama2 and Zephyr consistently provided high-quality answers, Mistral occasionally struggled with accuracy, leading to hallucinations. This evaluation underscored the importance of selecting the right model based on the specific requirements of accuracy, response time, and reliability.

Despite the success of this project, several areas offer opportunities for future improvements. These include the inclusion of new data files to enhance the model's knowledge base, model expansion to improve performance and accuracy, and advanced user interface features. Specifically, saving and reloading previous conversations would allow for continuity in user interactions, and the creation of user profiles would enable better customization based on individual preferences and past interactions, ultimately leading to a more personalized and effective user experience.

References

[1] Ollama: Ollama. https://ollama.com/

[2] OpenAI: ChatGPT. https://www.openai.com/chatgpt

[3] Uppal, C.: Legal tech transformation: How ai chatbots are revolutionizing access to justice (2023)

[4] Association, A.B.: Chatbots and the business of law (2023)

[5] Institute, T.R.: There's potential for ai chatbots to increase access to justice (2023)

[6] Diario da República: Diario da República. https://diariodarepublica.pt/dr/home

[7] Chroma: ChromaDB. https://www.trychroma.com/

[8] LangChain: LangChain. https://www.langchain.com/

[9] Streamlit: Streamlit. https://streamlit.io/

[10] PyPDF2: PyPDF2. https://pypdf2.readthedocs.io/en/3.x/

[11] PyTesseract: PyTesseract. https://pypi.org/project/pytesseract/

[12] PDFPlumber: PDFPlumber. https://pypi.org/project/pdfplumber/

[13] Google: GoogleTrans. https://pypi.org/project/googletrans/

[14] Translate. https://pypi.org/project/translate/

[15] Translators https://pypi.org/project/translators/

[16] DeepTranslator. https://deep-translator.readthedocs.io/en/latest/

Attachments

Graphs

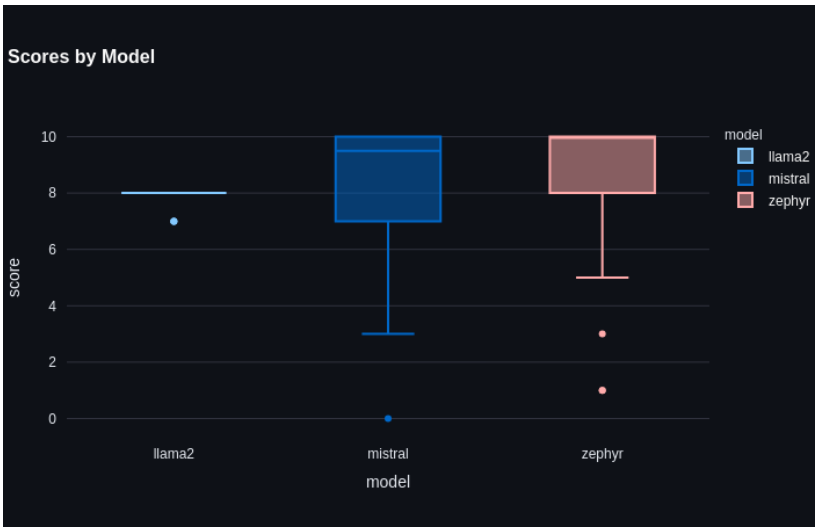

Fig. 6 Box Plot showcasing the variance in each model's scores

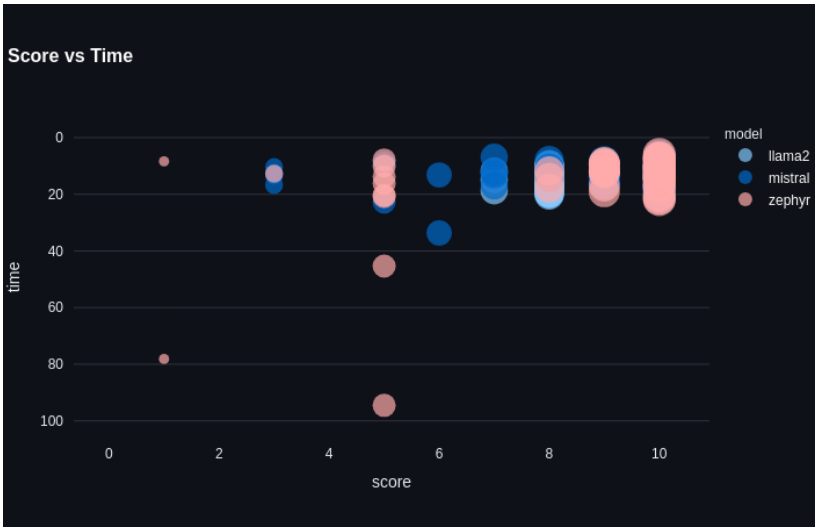

Fig. 7 Graph displaying the comparison between scores and generation time for each model

Prompt Examples

Fig. 8 Example of Llama2's output to a standard law question

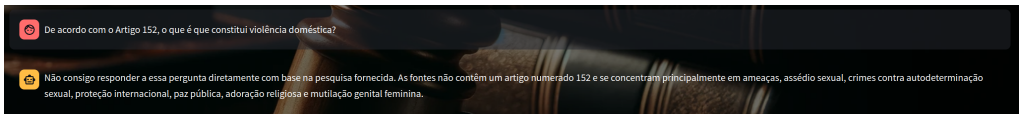

Fig. 9 Example of Mistral's output to a standard law question

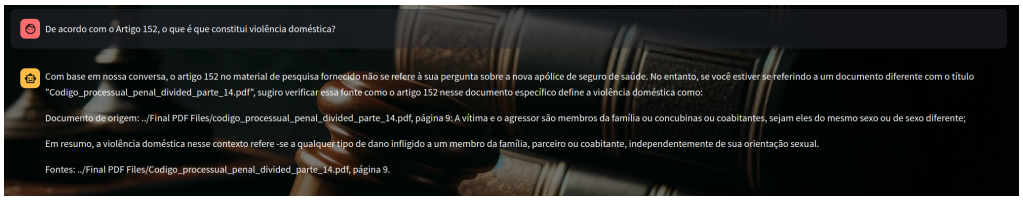

Fig. 10 Example of Zephyr's output to a standard law question