Employing Retrieval Augmented Generation to optimize LLMs for the legal domain

Lorenzo Schumann

December 2023

Employing Retrieval Augmented Generation to optimize LLMs for the legal domain: Evaluating methods to improve chat-bot performance.

1 Abstract

This paper explores the application of Large Language Models (LLMs) in the legal domain, utilizing Retrieval Augmented Generation (RAG) to optimize the performance of Llama 2. First, we demonstrate that integrating RAG with various prompting methods and a classification step significantly enhances Llama 2-Chat's effectiveness. Furthermore, we show RAG's capability in document selection and ranking, proving its utility in legal document analysis. Our findings affirm the potential and merit of employing LLMs in legal settings. The study also opens avenues for future research, including Query Expansion, further integration of ranking models with chatbots, combining LLMs with tools, and exploring hybrid retrieval mechanisms.

Keywords: *Large Language Models · Retrieval Augmented Generation · Document Ranking · Llama 2 · Prompt Engineering · LangChain · Fine-tuning · Legal AI*

Acknowledgements: Recognition is given to Professor Qiwei Han for his persistent support and guidance and to Professor Fabrizio Esposito for his valuable suggestions during the thesis work.

Table of Contents

1 Abstract **1**

2 Introduction **4**

3 Motivation **5**

4 Literature Review **6**

4.1 Legal AI and LLM Applications 6

4.2 Bias and LLM Data 9

5 Large Language Models **10**

5.1 Alternatives 10

5.2 Llama 2 11

5.3 Accessing Llama 2 13

5.4 Deploying Llama 2 14

5.5 Prompting Llama 2 14

6 Retrieval Augmented Generation (RAG) **15**

6.1 Overview 15

6.2 Comparison to Extraction Models 15

6.3 Effectiveness of RAG in Various Domains 16

6.4 Challenges and Limitations 16

6.5 Implementing RAG with LangChain 17

6.6 Working with Long Contexts 19

7 Methods to Improve Chatbot Performance **20**

7.1 Introduction 20

7.2 Methodology 20
7.3 Model Variations 21
7.4 Rationale Model Variations 22
7.5 Prompting Methods 22
7.6 Rationale for Prompting Methods 24
7.7 Results and Analysis 24
7.8 Comparative Analysis 27
7.9 Discussion 32
7.10 Conclusion 35

8 Group Conclusion **36**

References **37**

9 Appendix **45**

2 Introduction

Over the past decade, machine learning (ML) and natural language processing (NLP) domains have produced various language models (LMs) with increasing complexity and performance (Paaß and Giesselbach, 2023; Katz et al., 2023). Greater computational capabilities, better ML algorithms, the improved availability of big data sets, and the reduced cost of data storage led to the rise of large language models (LLMs). Compared to traditional LMs, LLMs are trained on far greater amounts of training data from various domains (Zhao et al., 2023), allowing them to answer complex questions, perform tasks in multiple languages, and generate outputs that blur the lines between human-written and machine-generated texts (Hou et al., 2023). Following the publication of OpenAI's ChatGPT 3.5 in November 2022, publications on generative artificial intelligence and LLMs surged in popularity within industries and academia (Zhao et al., 2023). Large tech companies like Meta and Google quickly followed up with their own LLMs, introducing various open-source models and creating a playground for developers and users alike. Today's landscape of custom LLMs and training tools is diverse, and it continues to grow (Naveed et al., 2023). Our project is focused on adapting an open-source large language model to function effectively in the legal domain, aiming to tailor it for retrieval and ranking tasks specific to legal texts and to operate as a specialized chatbot within this field. Our work does not intend to deliver a "ready-to-use product" but merely a proof of concept for the value proposition LLMS holds for the legal sector.

This thesis follows a clear structure to facilitate the reader's understanding of the tools we employed during our work and the reasoning behind our approach. Initially, we provide a brief motivation to explain why this topic is relevant for the legal domain, employing the example of price personalization. Then follows a literature review on legal artificial intelligence. We continue with a section on the most important models and methods applied in our work, such as the Llama 2-Chat model by Meta and the concept of Retrieval Augmented Generation (RAG). Subsequently, we move into the individual parts: Lorenzo's part about methods to improve chatbot performance and Anton's part about retrieving and ranking relevant documents. Finally, we end the thesis with

a brief conclusion.

3 Motivation

Several underlying factors drive the motivation for developing custom LLMs in the legal domain. When enhanced with RAG, LLMs can expedite workflows by providing relevant information. The integration of RAG and local LLMs tackles a primary obstacle, the concerns surrounding data protection and privacy, especially present with legal work utilizing publicly accessible LLMs Charlotin (2023*a*). As we will delve into in the latter section, RAG provides a cost-effective method for customizing a local or private LLM. This customization is crucial as it enables the selective integration or exclusion of data sources from the external database, thus aligning more closely with legal industry requirements.

Moreover, our research extends beyond just the implementation of RAG. We are dedicated to optimizing the performance of local LLMs, with a specific focus on enhancing chatbot functionalities. Furthermore, we investigate various improvement strategies, such as fine-tuning LLMs and exploring advanced document ranking techniques. Collectively, these efforts aim to tailor LLMs better to serve the nuanced needs of the legal sector, ensuring they are both effective and compliant with industry standards.

4 Literature Review

Before discussing our selected approach and tools in more detail, we use this section to provide the reader with a better understanding of the academic context of our work. Initially, we briefly introduce legal artificial intelligence (legal AI) (Katz et al., 2023; Wu et al., 2023) and its various applications. Next, we dive deeper into the role of language models LMs and LLMs in legal AI, presenting several prominent publications. Subsequently, we briefly discuss Zero-shot, Few-shot, and chain-of-thought prompting before moving on to review the role of data bias in an LLM context.

4.1 Legal AI and LLM Applications

Legal AI encompasses all machine learning techniques and models aimed at enhancing and facilitating processes in the legal domain (Zhong et al., 2020; Katz et al., 2023; Wu et al., 2023). Over the past decade, researchers developed a variety of legal AI applications with the potential to change the future of legal work as we know it today (Charlotin, 2023*b*; Katz et al., 2023), reaching from NLP models for legal language understanding (Chalkidis et al., 2022; Nawar et al., 2022), over the summarization of legal texts (Bhattacharya et al., 2019; Jain, Borah and Biswas, 2021), legal question answering (Xiao et al., 2021; Do et al., 2017), and judgment prediction (Strickson and De La Iglesia, 2020; Xiao et al., 2021; Paul et al., 2023; Wu et al., 2023) all the way to court view generation (Ye et al., 2018; Wu et al., 2020) and more (Wu et al., 2023). The introduction of LLMs, often referred to as foundational models due to their broad basis of training data with minor specialization (Bommasani et al., 2022), provided new direction and opportunities to legal AI, offering researchers an adjusted toolkit of entirely new dimensions (Lai et al., 2023).

4.1.1 Custom Legal Language Models

Between the many applications of AI in law, the potential of LMs and LLMs in legal research is of particular interest to our work. Although our work revolves around the customization of an LLM,

and there are differences between LLMs and LMs, most of the scope of application for LMs could be covered by an LLM (Zhao et al., 2023). We want to utilize the following section to provide an overview of a few custom legal LMs, pre-trained language models (PLM), and LLMs and their applications.

One of the first PLMs designed explicitly for processing long legal documents was "Longformer," proposed by Xiao et al. (2021). Their approach was new in that they did not apply a full self-attention mechanism but instead used a combination of local sliding window attention and global task-motivated full attention to capture dependencies that lay further apart. Combining techniques allows the "Longformer" model to encode documents with thousands of tokens, a challenge that previous models were not able to tackle and provided a significant hurdle for legal LMs (Xiao et al., 2021). Niklaus and Giofré (2022) continued to build on the "Longformer" transformer model to create a low-budget but state-of-the-art LLM. The authors employed the "replaced token detection task," a method that helps LMs to learn word context by guessing the replaced token to reduce computational costs and managed to create a capable summarization model.

Two prominent recent LLMs include "Lawyer LLaMA" by Huang et al. (2023) and "ChatLaw" by Cui et al. (2023). To create "Lawyer LLaMA" Huang et al. (2023) developed an advanced framework to adapt LLMs to domain-specific knowledge throughout a continuous training process and added a document retrieval mechanism to mitigate the common problem of hallucinating LLMs. This style of retrieval mechanism is very similar to the process for retrieval used by RAG. We will elaborate on this mechanism in a later section. While their work mostly focused on legal information regarding marriage-related problems, they still created a framework with large potential for other areas (Huang et al., 2023). The openly available versions of "Lawyer LLaMA" are capable of providing legal advice, analyzing legal cases, and generating legal articles (Lai et al., 2023). Similarly, "ChatLaw" was introduced by Cui et al. (2023) at Beijing University. Like Huang et al. (2023), Cui et al. (2023) employed a vector database retrieval mechanism but enhanced it with a keyword retrieval mechanism that allowed for the further reduction of inaccuracies. Furthermore, Cui et al. (2023) introduces a self-attention mechanism to combat hallucinations.

4.1.2 Emerging Abilities and Scaling

While general LMs and LLMs are no new concepts, Zhao et al. (2023) rightly points out that recent advances have led to an evolution of LMs, becoming capable of solving tasks of far greater complexity. Among the drivers of development, only second to the introduction of transformer architectures, which revolutionized the field of NLP by providing a more efficient, scalable, and effective way to process and understand language (Han et al., 2021; Zhao et al., 2023; Katz et al., 2023), comes the scaling of models and their so-called ”emergent abilities” (Rosa et al., 2022; Wei et al., 2022).

Emergent abilities are those abilities that arise with model size and do not underlie strict scaling laws (Wei et al., 2022). The most frequently discussed emergent abilities are Zero-shot or Few-shot reasoning, i.e., tasks for which a model is prompted with no examples or, in the latter case, only a few examples (e.g., Brown et al. (2020)). LLMs' Few-shot learning capabilities were first pointed out by Brown et al. (2020), demonstrating that Few-shot prompting could enable performances on par with those that were previously only achieved by state-of-the-art fine-tuning approaches (Brown et al., 2020). About two years later, Wei et al. (2022) published a paper introducing the notion of chain-of-thought prompting, a method of dividing tasks into a series of intermediate reasoning steps for the model to complete that enabled LLMs to perform tasks of even greater complexity than before. Their results showed potential for areas in arithmetic, symbolic, and common sense reasoning and have since led to the wide adoption of chain-of-thought prompting in the academic community (see: Wang et al. (2022); Wei et al. (2022).

Since the introduction of the notions of Few-shot prompting by Brown et al. (2020) and chain-of-thought prompting by Wei et al. (2022), employing combinations of the three prompting styles (including Zero-shot prompting) has become common practice for performance evaluation (e.g., Zhou et al. (2023); Blair-Stanek, Holzenberger and Van Durme (2023), and researchers continue to explore their use cases in various legal contexts (e.g., Rosa et al. (2022); Blair-Stanek, Holzenberger and Van Durme (2023); Zhou et al. (2023)). To give an example, take Rosa et al. (2022) who followed up on previous research regarding the Zero-shot capabilities in legal case entailment

tasks (a task to identify paragraphs in one legal case that entail decisions in another (Li et al., 2023*a*)) by Rabelo, Kim and Goebel (2021) and similar work on the research of scaling effects to experiment with Zero and Few-shot performance. Their work showcased the potential of LLMs in legal entailment tasks and proved once again how larger LLMs, meaning models with more parameters, show great performance improvements in Zero-shot and Few-shot contexts, so much so that models with greater parameter count and little to no domain-specific training data can outperform smaller models containing domain training data (Rosa et al., 2022). Other examples of research on legal AI, including the aforementioned emerging capabilities, include the investigation of GPT's ability to perform statutory reasoning (Blair-Stanek, Holzenberger and Van Durme, 2023), the prediction of relevant law articles for legal cases (Chen, Chiang and Wu, 2022), and the classification of controversial issues in legal documents (Fang et al., 2020).

4.2 Bias and LLM Data

The rapid advancement of LLMs like GPT-3 presents a complex landscape of ethical challenges, particularly in the realms of bias and discrimination Sun (2023). While LLMs carry vast transformative potential across sectors, researchers repeatedly point out that they carry the risk of perpetuating and even amplifying biases present in their training data. For instance, racial bias in algorithms like COMPAS Brackey (2019) and anti-queer bias in models like BERT Felkner et al. (2022) highlights the central problem of algorithmic discrimination. These biases, rooted in historical and systemic inequalities, can lead to discriminatory outcomes, especially in sensitive applications like legal judgments and societal analysis Lai et al. (2023). Although approaches to battling bias in language models exist, for intrinsic bias, applicable during pre-processing (e.g., (Zmigrod et al., 2019; Stahl, Spliethöver and Wachsmuth, 2022)), in-processing (e.g., (Webster et al., 2020; Ahn and Oh, 2021)), and post-processing phase of LLMs, and for extrinsic bias (e.g., Dixon et al.dixon2018measuring Li et al. (2023*b*); Panda et al., panda2022don) bias continues to be an active problem. Investigations by researchers (Hemmatian, Baltaji and Varshney, 2023) how the effects of de-biasing can be diminished through continued model development.

5 Large Language Models

5.1 Alternatives

LLMs are categorized either categorized as open and closed models. Open models, such as BLOOM, LLaMa, and Falcon, are available for public use and modification, offering transparency and collaborative potential for the research community. On the other hand, closed models are proprietary and often restricted in terms of access and adaptability. Among the closed models, "product" LLMs like ChatGPT, BARD, and Claude have undergone extensive fine-tuning to align closely with human preferences, enhancing their usability and safety for end-users. These product LLMs typically outperform open models regarding user interaction quality, as they are optimized through substantial computational resources and human annotation (Touvron et al., 2023).

For research purposes, however, open-source models are more valuable. They allow for reproducibility, transparency in methodology, and the ability to build upon previous work, which is crucial for advancing the field. Open-source models enable researchers to explore the inner workings of these complex systems, experiment with modifications, and understand the impact of various training techniques, which is often impossible with closed models. (Liu et al., 2023*b*). Furthermore, a comprehensive overview of open LLMs is available on GitHub, facilitating informed decision-making for integrating language models into commercial products and services (Yan, 2023).

5.1.1 Rationale for choosing Llama 2

The two main reasons for choosing Llama 2 over other LLMs are its open-source status and performance benchmarks. Llama 2's designation as an open-source model significantly enhances its appeal in academic and practical applications by offering unrestricted access to its foundational technology. Regarding performance, Llama 2 stands out for its competitive results, particularly in key areas such as helpfulness and safety, demonstrating capabilities that rival or even exceed those

of closed-source models (Touvron et al., 2023). This combination of open accessibility and robust performance makes Llama 2 a compelling choice.

5.2 Llama 2

Llama 2 is a collection of LLMs encompassing a range of models with parameters varying from 7 billion to 70 billion. The collection includes Llama 2-Chat, fine-tuned models specifically optimized for dialogue applications (Touvron et al., 2023).

5.2.1 Llama 2 Base model

Using standard evaluation benchmarks, Llama 2 base models demonstrate a performance that surpasses that of many other models with comparable sizes. Additionally, there is a positive relationship between model size and performance. This trend suggests that the larger Llama 2 models are well-suited to handle more complex and nuanced tasks, positioning them as strong contenders for academic research and practical AI applications. However, even the smaller models display strong performance metrics, establishing them as viable options for research-focused activities (Touvron et al., 2023).

5.2.2 Llama 2-Chat model

The chat version of Llama 2 is developed in three steps. First comes the pre-taining of the base model. Then, an initial version of the chat model is created using supervised fine-tuning. Finally, the model is refined using Reinforcement Learning with Human Feedback (RLHF) methods, in this case, rejection sampling and Proximal Policy Optimization (PPO) (Touvron et al., 2023).

5.2.2.1 Reinforcement Learning with Human Feedback

After the initial supervised fine-tuning, RLHF is used to align the model further with human preferences. This method commences by creating human preference data, where annotators create a

prompt and receive two model-generated responses, varying in temperature and hyperparameters. The annotator selects the preferable response and rates the degree of its superiority over the alternative. This preference data then informs the training of a reward model, which evaluates a combination of the prompt, model response, and historical context, producing a scalar score that reflects the output's quality. Meta employed two reward models, one for safety and one for helpfulness. Subsequently, PPO and rejection sampling are employed to refine the model based on the reward model's feedback. PPO methodically updates the model's policy to optimize rewards, ensuring minimal deviation from the previous policy for consistent training. Rejection sampling complements this by generating multiple responses and selecting the responses with the highest quality as per the reward scores. This approach guarantees the continuation of only the most effective responses, progressively elevating the model's performance. The refined model is then used to generate new human preference data sets. This cycle of generating preference data, refining the model, and employing the updated model for further data generation is repeated multiple times, each iteration enhancing the reward model and, consequently, the overall effectiveness of Llama 2-Chat (Touvron et al., 2023).

5.2.2.2 Evaluation of Llama 2-Chat

As noted by Touvron et al. (2023), determining the best practices for evaluating an LLM remains a topic of ongoing research, with human evaluation currently considered the most reliable standard. Despite this, an enhancement in the reward model is directly linked to improvements in the chat model's performance. In their approach to determining the reward model's accuracy, the researchers set aside 1000 examples each time they collected human preference data. They also incorporated publicly available alternatives to establish a comparative baseline. The reward models underwent evaluation focused on helpfulness and safety, achieving scores of 70.4 and 64.3, respectively. Notably, this performance surpasses that of GPT4, underscoring the high quality of Llama 2-Chat (Touvron et al., 2023).

5.2.3 Rationale for Using the Chat-Version

As highlighted earlier, aligning the baseline model more closely with human preferences and adapting it for chat interactions demands considerable effort and resources, exceeding the scope of our research. Therefore, opting to use a pre-tuned chat model rather than fine-tuning the base model ourselves enables a more direct focus on achieving the paper's objectives.

5.2.4 Rationale for Using the 7B Version

Considering the resource constraints, the 7B model emerges as the practical choice. Its relatively smaller size offers faster processing, albeit with a trade-off in accuracy (Touvron et al., 2023). Attempts to load the 13B model proved unsuccessful due to the limitations of the available server capacity. Thus, the 7B model is the sole viable option under the given circumstances. However, future research endeavors are encouraged to explore the larger models to reassess and potentially enhance the findings.

5.3 Accessing Llama 2

Accessing Llama 2 requires the completion of a form on the official webpage (Meta AI, 2023). Once access is granted, there are two ways to download Llama 2. The first method involves following the guidelines provided on Meta's GitHub profile. Alternatively, a simpler approach is to load the model directly from Huggingface. For this, one must request access again on Huggingface using the same email registered on the Meta webpage. After access is approved, the model can be downloaded using the *login* package from $huggingface\_hub$ or cloning the repository directly from Huggingface.

5.4 Deploying Llama 2

Several deployment methods are available for Llama 2. One approach involves utilizing the *AutoTokenizer* and *pipeline* from the *tokenizer* package. Alternatively, a more user-friendly option with an integrated interface is available through the LMFlow GitHub page (Diao et al., 2023).

5.5 Prompting Llama 2

Prompting the Llama 2-Chat model requires using a specific format to achieve good results. The prompting format for chatbot interactions with a system message can be seen in Figure 1.

```
<s>[INST] <<SYS>>
{your_system_message}
<</SYS>>
{user_message_1} [/INST]
```

Figure 1: Prompting Llama 2 - With System Message

When prompting without a system message, it is crucial to adhere to the format displayed in Figure 2. Deviating from these formats can significantly diminish the quality of the output. While the default system message instructs the chatbot to maintain respectfulness (Utils, 2023), replacing this default with context-specific information or chat history is possible.

```
<s>[INST] {user_message_1} [/INST]
```

Figure 2: Prompting Llama 2 - Without System Message

6 Retrieval Augmented Generation (RAG)

6.1 Overview

RAG, developed by Lewis et al. (2020), marks a crucial advancement in language model technology, particularly for tasks requiring comprehensive external knowledge. Despite their proficiency across various tasks, traditional LLMs often encounter challenges in dealing with knowledge-intensive queries. RAG addresses this shortfall by merging an information retrieval component with a sophisticated text generation model. This combination substantially enhances the factual accuracy and overall reliability of the responses generated. In LLMs, parametric memory is the information ingrained within the model's parameters, creating an implicit knowledge base through extensive training on diverse datasets. This intrinsic knowledge enables the models to generate informed responses, but they struggle with updating their knowledge, clarifying the reasoning behind their outputs (Lewis et al., 2020), and can produce hallucinations (Azamfirei, Kudchadkar and Fackler, 2023). Non-parametric memory, in contrast, involves accessing external, explicit knowledge sources, such as databases, allowing the model to retrieve and integrate up-to-date information, thus enhancing its responses. RAG combines these two forms of memory, utilizing a neural pre-trained retriever for non-parametric memory access. This configuration allows the models to remain up-to-date and accurate, overcoming the traditional constraints of static knowledge (Lewis et al., 2020).

6.2 Comparison to Extraction Models

RAG outperforms other models in handling a diverse range of question types by blending the generative strengths of closed-book models with the comprehensive retrieval abilities of open-book approaches. Instead of relying on complex pre-training, RAG uses a dynamic retrieval approach that pulls from an external knowledge base to support its internal, parametric knowledge. This approach leverages contextual cues within source documents for answer generation, utilizing in-

formation even when the explicit answer isn't directly present in the documents, enabling RAG to produce correct answers in scenarios where traditional extractive models would fail. In instances where the answer isn't directly found in the documents, RAG demonstrates 11.8% accuracy, highlighting its effectiveness compared to extractive models that would yield no correct responses in these situations. This capability marks a significant advantage over conventional models, positioning RAG as a more versatile and up-to-date tool for knowledge-intensive applications (Lewis et al., 2020).

6.3 Effectiveness of RAG in Various Domains

RAG enhances the utility of LLMs by integrating their broad knowledge bases with specialized domain-specific information. This unique capability significantly boosts the LLMs' effectiveness across various fields. Domains that demand detailed knowledge, which might not be covered by the LLMs inherent parametric knowledge, particularly gain from RAG's approach (Lewis et al., 2020). RAG's ability to incorporate domain-specific information into LLMs could prove invaluable in the legal domain, where practitioners frequently engage with unique client documents and specialized research.

6.4 Challenges and Limitations

While RAG models have shown remarkable capabilities in enhancing LLMs, they are not without limitations. Significant challenges lie in their inherent complexity and the need to handle long inputs. These limitations require the model to manage extensive data efficiently. Moreover, there is an evident need for the integration of re-ranking mechanisms within RAG models. These mechanisms are vital in refining the accuracy of results, ensuring that the most relevant and precise information is retrieved from the knowledge base. This aspect is crucial for maintaining the overall effectiveness and reliability of RAG models in various applications (Cai et al., 2022).

6.5 Implementing RAG with LangChain

LangChain provides specialized components designed for the development of RAG applications. The process of developing a RAG application encompasses five steps: loading data, splitting the text data into chunks, embedding the chunks and saving them in a vector store, retrieving relevant chunks based on the query, and generating an output with an LLM using the retrieved context (LangChain, Inc., 2023*d*). While many document loaders, transformers, vector stores, embedding models, and chat models exist, the ensuing section will concentrate on the general process and not emphasize specific tools.

6.5.1 Loading the Data

This step retrieves data from many sources, encompassing webpages, text files, databases, and other digital formats. The primary goal is to transform this acquired data into a uniform, simple text format. This process is crucial for the subsequent stages of the application, which rely on text-based analysis and processing (LangChain, Inc., 2023*d*).

6.5.2 Splitting the Data

Once the data is loaded and converted into plain text, it is essential to split it into smaller segments before storing it in the vector store. Text splitters serve this purpose effectively. They ensure the data is broken down into manageable parts, facilitating efficient storage and retrieval. Furthermore, this division is beneficial for data indexing, making it more searchable, and also crucial for effectively feeding the data into a model. Given the finite context window of most models, large chunks of text can pose challenges in processing and analysis. Smaller segments, however, can be more easily navigated, searched, and fit within a model's processing capabilities. This step ensures that the data is optimally prepared for detailed examination and interaction with the model, enhancing the overall efficiency and accuracy of the system (LangChain, Inc., 2023*d*).

6.5.3 Storing Data in Vector Form

After splitting the text into smaller chunks, the next step involves storing and indexing these segments for efficient retrieval. This task is typically accomplished using a vector store combined with an Embedding model. The vector store functions as a database, organizing and maintaining the text splits, while the Embedding model transforms these text segments into vector representations. These vectors are then indexed within the vector store, enabling rapid and precise searching capabilities. This process is fundamental for quick access and retrieval of relevant information. Additionally, relevant metadata can be stored for later use (LangChain, Inc., 2023*d*).

6.5.4 Retrieval Process

When a user input is provided, the system initiates a retrieval process using a Retriever component. This Retriever is designed to search the stored data efficiently, specifically, the text splits within the VectorStore. By leveraging the vector representations of these splits, the Retriever can quickly identify and fetch the most relevant segments of text that match the user's query. A wide range of similarity metrics and scoring systems are available to customize the functionality of the Retriever. These tools allow for fine-tuning the Retriever's performance to align with specific needs and objectives (LangChain, Inc., 2023*d*).

6.5.5 Generation with RAG

In the final step of the process, a Chat Model or LLM generates a response. This response is based on a prompt that integrates both the original user question and the data retrieved in the previous steps. Incorporating retrieved data ensures that the answer produced is not only contextually relevant but also enriched with specific information pertinent to the query, enhancing the overall accuracy and informativeness of the model's output (LangChain, Inc., 2023*d*).

6.5.5.1 Adding Context to Llama 2

When integrating Llama 2 with context, it's crucial to include this context within the system message (Figure 1). The system message is a key component that guides the model in understanding the context of the conversation or the query. By embedding the necessary context directly into the system message, Llama 2 is better equipped to provide relevant and accurate responses.

6.6 Working with Long Contexts

In LLMs, especially when handling long contexts with over ten retrieved documents, there's a significant issue where information in the middle of the context often gets overlooked. To mitigate this, a strategic approach involves reordering the documents after retrieval. This reorganization prioritizes the most relevant information, ensuring it's positioned at the beginning and end (Liu et al., 2023*a*). *LongContextReorder* from LangChain can be used to perform the reorganization (LangChain, Inc., 2023*c*).

7 Methods to Improve Chatbot Performance

7.1 Introduction

This study evaluates various strategies to enhance chatbot performance. The primary aim was to investigate the influence of RAG and diverse prompting methods on the performance of chatbots, with a special emphasis on how these techniques could augment the models' capabilities in handling questions about the domain of price personalization. The research adopted a comprehensive methodology involving data collection, storage, and, subsequently, the use of RAG in combination with various prompting methods. A structured questionnaire was employed to evaluate the performance of these models, covering both general and specific aspects of price personalization and closely related areas. The key findings highlight the effectiveness of RAG in boosting LLM performance and reveal the varying impacts of different prompting methods on model accuracy. Additionally, the research underscored the value of integrating filtering or classification steps when using LLMs with RAG, which notably enhanced their proficiency in dealing with detailed, context-specific questions, thus demonstrating the potential of this approach in refining the performance of LLMs in real-world applications.

7.2 Methodology

7.2.1 Data Collection and Preparation

To create a dataset, co-advisor Professor Fabrizio Esposito from the Faculty of Law, Universidade Nova de Lisboa, recommended using the top ten pages from Google Scholar when searching "price personalization AND law" and "personalized price AND law." This gives a comprehensive collection of papers relevant to price personalization. The papers can be downloaded in PDF format and are stored locally. Using *PyPDFLoader* from LangChain, these documents can be loaded in a format that allows subsequent processing and storage in a vector database.

For RAG to work accurately, the data needs to be prepared by dividing the texts into chunks and ensuring they have a consistent format. In this step, it is crucial to perform the formatting after the chunking because the *RecursiveCharacterTextSplitter* from LangChain uses separators such as newlines and similar expressions to create chunks that have a semantic connection. These separators are removed in the subsequent formatting step. The resulting chunks are then stored with relevant metadata, including the publisher, title, author, and citation, using *Huggingface Embeddings*. (For more information on vectorization and storage, please refer to section 6.5.3).

7.3 Model Variations

Five different variations of the 7B Llama-2-Chat model will be evaluated. Initially, a model without any alterations, then two variations, respectively, using RAG with cosine similarity and maximum marginal relevance (MMR). The final two versions once again employ cosine similarity and MMR, this time filtering documents before initializing RAG.

7.3.1 Cosine Similarity and Maximum Marginal Relevance

RAG retrieves the most relevant chunks stored in the database, as explained in section 6.5.4. To perform vector retrieval, both cosine similarity and MMR can be employed. Cosine similarity selects strictly the most relevant chunks by similarity to the query (Rahutomo, Kitasuka and Aritsugi, 2012). MMR, on the other hand, chooses the document that maximizes a combination of both relevance and diversity. This is particularly useful in scenarios where users might want a wide range of information on a topic rather than multiple similar documents or answers that all say essentially the same thing (Carbonell and Goldstein, 1998). To measure the difference in performance, both Cosine Similarity and MMR are separately evaluated during the testing.

7.3.2 Additional Filtering

A strategy presented by OpenAI to improve RAG performance is an additional classification step to more accurately select relevant embeddings (OpenAI, 2023). The filtering can be done automatically with a classification tool (OpenAI, 2023), a technique beneficial for users with limited database knowledge. Another option is to let the user select documents relevant to the query. Since I focus on users who have extensive knowledge in the legal domain, the testing will revolve around manual document selection. Nevertheless, determining the efficiency of automatic document selection in combination with a chatbot and RAG should be considered by future research.

7.4 Rationale Model Variations

Employing models without RAG or filtering is crucial to establish a baseline. The comparison with RAG models highlights RAG's potential to improve the performance of chatbots. Implementing cosine similarity and MMR as distinct similarity measures further reveals their varying impacts on model efficiency. Lastly, incorporating a filtering or classification step, which targets information directly related to the query, illustrates how focusing on relevant data sources can significantly refine the model's output and effectiveness. Therefore, using these five model variations provides a comprehensive analysis of the different aspects influencing model performance.

7.5 Prompting Methods

To further investigate methods to improve LLM performance, the prompting methods Zero-shot prompting, Few-shot prompting, Chain-of-Thought (CoT) prompting, and a variation of Self-consistency will be tested. These four methods will be evaluated with all five models, resulting in twenty different combinations.

7.5.1 Zero-Shot Prompting

Research has demonstrated that LLms can perform tasks ”Zero-shot” (Kojima et al., 2022). When models are given prompts that specify the expected action or response, the model can infer the correct approach by leveraging the broad knowledge base the model was originally trained on. This is especially useful when gathering a large, task-specific dataset is impractical. Furthermore, not including examples reduces token usage and, consequently, the costs per query (Kojima et al., 2022).

7.5.2 Few-Shot Prompting

LLMs have notable capabilities in Zero-shot tasks but tend to underperform when given more complex tasks within the same setting. Few-shot prompting is a technique that uses in-context learning by incorporating example demonstrations directly into the prompt. These demonstrations help condition the model for improved performance on subsequent tasks, where it is expected to generate responses based on the context provided by these examples (Brown et al., 2020). This paper adopts three-shot prompting as its methodology, incorporating three distinct examples. When using the Llama 2-Chat model, it is important to use the correct format to create these examples. The example questions and answers must be presented in this format: $[INST]$ *example question* $[/INST]$ *example answer*. Not using this format will result in the model interpreting the examples as questions and answering them as well.

7.5.3 Chain-of-Thought Prompting

Chain-of-thought (CoT) prompting was introduced by Wei et al. (2022). It enhances complex reasoning in LLMs by introducing intermediate reasoning steps. This approach, when combined with Few-shot prompting, significantly improves outcomes on complex tasks that require reasoning before generating a response (Wei et al., 2022). As in section 7.5.2, using the correct format to demonstrate the reasoning path is very important.

7.5.4 Self-Consistency

Self-consistency is a sophisticated prompt engineering technique introduced by Wei et al. (2022). The goal is to improve the basic greedy decoding approach commonly used in CoT prompting. The method involves generating multiple, varied reasoning paths by using Few-shot CoT prompting. These multiple paths are then analyzed to identify the most consistent answer. This technique enhances the effectiveness of CoT prompting, particularly in tasks that require arithmetic calculations and commonsense reasoning (Wei et al., 2022). Due to resource constraints, traditional Self-consistency is out of scope for this paper. Alternatively, selecting the most common answer among Zero-shot, Few-shot, and CoT prompting was employed. Nonetheless, I encourage further research to determine the performance of traditional Self-consistency in combination with the aforementioned models.

7.6 Rationale for Prompting Methods

Zero-shot prompting is crucial to gauge the fundamental abilities of large language models, establishing their base-level performance (Kojima et al., 2022). Few-shot prompting further reveals a model's capacity to learn from just a few examples, showcasing adaptability (Brown et al., 2020). CoT prompting specifically assesses reasoning skills by breaking down complex problems (Wei et al., 2022). Self-consistency, in contrast, focuses on the model's reliability and the consistency of its outputs (Wang et al., 2022). Employing these various prompting methods provides a comprehensive evaluation of the different facets and capabilities of the model variations.

7.7 Results and Analysis

7.7.1 Testing Approach

The models were tested with a questionnaire containing 60 questions about price personalization. To facilitate testing, the number of papers was reduced to one-fourth of the Papers presented in

section 7.2.1. The Questions encompass multiple choice (11), alternative-response questions (15), and open-ended questions (34) to assess the performance across a variety of question formats. Considering that the questions will mostly be open-ended in a real-world scenario, this category contains the most questions. Moreover, half of the questions are general questions about price personalization and closely related areas, and the other half are specific to the topics mentioned in the papers. This distribution was added to determine whether RAG retained its established knowledge and supplemented it with new insights. The model's output was assessed on a scale from 0 to 1, demonstrating the accuracy of the output compared to a predetermined optimal response. While it can be difficult to give an exact number on the accuracy of a response of this type (Touvron et al., 2023), special care was taken to ensure that the score is comparable between the different approaches, ensuring the comparability of the outcome. Hence, when looking at the results, the magnitude of accuracy is less important than the relative difference between scores.

7.7.2 Performance Evaluation

Figure 3 shows the outcome of the testing and demonstrates that RAG improved the base model performance significantly. An additional filtering step improved the performance even further. Both cosine similarity and MMR produce near-identical results for Self-consistency. Nevertheless, MMR performed slightly better with Zero-shot and Few-shot prompting. Cosine similarity, on the other hand, worked marginally better with CoT prompting. While all prompting methods notably increased the performance of the Zero-shot base model, the same can not be said for the RAG models. Few-shot prompting generally decreased or maintained the accuracy, RAG (MMR) with filtering being the only exception. CoT and Self-consistency, on the other hand, increased the performance of every model except for RAG (MMR).

Prompting Methods	Base Model	RAG (MMR)	RAG (Cosine Similarity)	RAG MMR & Filter	RAG Cosine & Filter
Zero-shot prompting	47,50%	74,58%	71,25%	84,58%	85,00%
Few-shot prompting	51,25%	70,83%	69,17%	88,33%	85,00%
Chain of Thought prompting	57,50%	73,33%	74,58%	85,83%	87,08%
Self-consistency	54,17%	72,92%	73,75%	86,25%	86,67%

Figure 3: All Questions

7.7.2.1 Performance Evaluation - General and Specific Questions

As outlined in section 7.7.1, the questionnaire is evenly divided between general questions about price personalization and specific questions related to the dataset used for contextualizing the models with RAG. This division helps assess RAG's influence on general question performance and its efficacy in providing the model with extra data for complex queries. As Figures 4 and 5 demonstrate, the baseline model achieves an average accuracy of roughly 75%, indicating its effectiveness in addressing general questions. The incorporation of RAG enhances this performance to approximately 86%. Implementing an additional filtering step boosts the average accuracy to around 90%, indicating that when using RAG, the model not only retains its general knowledge but also expands it. However, the baseline model's mean accuracy of 29.79% for specific questions highlights a significant drop in performance when moving away from general knowledge. The introduction of RAG improves accuracy to around 59%, nearly doubling it. Introducing a filtering step further elevates this accuracy to over 80%. This improvement underscores the substantial impact of RAG, yet it also emphasizes the necessity of an additional filtering step for answering document-specific queries. As noted in section 7.3.2, this can be achieved through an expert user who filters for relevant documents or a ranking tool.

Prompting Methods	Base Model	RAG (MMR)	RAG (Cosine Similarity)	RAG MMR & Filter	RAG Cosine & Filter
Zero-shot prompting	74,17%	90,00%	83,33%	91,67%	91,67%
Few-shot prompting	67,50%	85,00%	83,33%	89,17%	90,00%
Chain of Thought prompting	80,00%	85,83%	89,17%	88,33%	94,17%
Self-consistency	80,00%	85,83%	89,17%	91,67%	90,83%

Figure 4: General Questions Results

Prompting Methods	Base Model	RAG (MMR)	RAG (Cosine Similarity)	RAG MMR & Filter	RAG Cosine & Filter
Zero-shot prompting	20,83%	59,17%	59,17%	77,50%	78,33%
Few-shot prompting	35,00%	56,67%	55,00%	87,50%	80,00%
Chain of Thought prompting	35,00%	60,83%	60,00%	83,33%	80,00%
Self-consistency	28,33%	60,00%	58,33%	80,83%	82,50%

Figure 5: Specific Questions Results

7.8 Comparative Analysis

The following section contrasts and compares the output of the different models and prompting methods. The goal of this section is to understand how both model variations and Prompting methods impacted the accuracy of the output. This assessment measures not only overall accuracy but also breaks down accuracy by question type and in general and specific categories, offering a fuller picture of where the various approaches excel and where they fall short. All presented outcomes are available in figures 8 to 16 in the appendix.

7.8.1 Baseline Model

The Baseline model recorded an average accuracy of 52.6%, the lowest among all tested models. CoT and Self-consistency emerged as the top performers, with CoT enhancing model performance by 10% over Zero-shot prompting. Few-shot prompting also yielded an improvement, though it was relatively minor. Regarding question categories, the model showed greater proficiency in general questions, achieving an accuracy of 75.42%. However, it struggled with specific questions, managing only a 29.79% average score. This disparity indicates that while the model possesses substantial general knowledge, it lacks the expertise required for more specialized queries. Looking at the performance per question type, it becomes clear that the baseline model is best at answering alternative-response questions, followed by multiple-choice and open-ended questions. This suggests the model's capability to create a reasoning pathway autonomously is relatively limited. This

outcome may be attributed to the deployment of the 7B model. Consequently, it is advisable to reevaluate the findings using an alternative model, such as the 70B Llama 2 model, for comparison.

7.8.2 Impact of RAG and Filtering

Above all, RAG and filtering impacted the accuracy of the model output the most. The RAG models improved the output by 19.95% compared to the base model's average accuracy. The additional filtering enhanced the regular RAG models by another 13.54%. While the mean difference between the MMR and cosine similarity models was less than 1%, with MMR having a slight edge with and without filters (0.31% and 0.73%, respectively), the performance for each question type varied significantly.

7.8.2.1 RAG with MMR

Using RAG combined with MMR results in an accuracy close to 70% for each prompting method. The mean accuracy for general questions is 86.67% and 59.17% for specific questions. This shows that RAG enhanced the general knowledge of the model and almost doubled the specific knowledge. Nevertheless, the accuracy of specific questions is quite low. While different prompting methods increased the performance of all the other models, Zero-shot prompting performed best for this model. RAG (MMR) worked particularly well for open-ended questions regardless of the prompting method, with Few-shot, CoT, and Self-consistency improving the outcome produced with Zero-shot prompting. The reason for a performance decrease with new prompting methods was a sharp performance decrease for alternative-response and especially multiple-choice questions. With new prompting methods, the average score of multiple-choice questions decreased by 9.9% and for alternative-response questions by 4.6%. This highlights the importance of selecting the appropriate prompting approach depending on the question type.

7.8.2.2 RAG with Cosine Similarity

The RAG model with cosine similarity results in an accuracy of around 70% for each prompting method, just as RAG with MMR. The performance for general and specific questions is also nearly identical. Nevertheless, the best-performing question category is not open-ended but multiple choice questions, followed by alternative-response questions and then open-ended. Hence, using a different similarity measure greatly impacts the question type the LLM can answer best. It is also noteworthy that the 91.91% accuracy for Zero-shot prompting is the highest performance across all models for multiple-choice questions, tied with the RAG (cosine similarity) with a filter. When new prompting methods are introduced, there's a decrease in the accuracy for Multiple Choice questions from this high of 90.91%. In contrast, the accuracy of open-ended questions increases with new methods, suggesting that these question types benefit from the additional context or structured reasoning these methods provide. For alternative-response questions, the accuracy remains relatively consistent across different prompting methods, except for Few-shot prompting, where there's a drop in performance to 71.43%, which is the only significant deviation from the model's generally stable performance in this question type.

7.8.2.3 RAG with MMR and Filter

Adding filtering to RAG (MMR) leads to an average accuracy of 86.25%. This is the highest compared to all other models. The accuracy for general questions saw a modest enhancement of less than 5%. However, the most significant progress was observed in the specific questions, with an improvement of 23.13%. This notable enhancement underscores the value of incorporating an extra classification phase when posing document-specific queries to the model. Examining the results by question type, the high-performance pattern on open-ended questions and lower scores on multiple-choice questions persists, particularly with Few-shot and CoT prompting, which yield the highest scores for open-ended questions across all model variations. Alternative-response questions achieve a score that is about 5% higher than multiple-choice questions and approximately 5% lower than open-ended questions on average. Nonetheless, employing Few-shot prompting leads to the

highest overall accuracy in alternative-response questions compared to all other models.

7.8.2.4 RAG with Cosine Similarity and Filter

The model using cosine similarity with a filter has an average accuracy close to 86%, which is inconsiderably different from the model using MMR with a filter. While the score for general questions is slightly higher and the score for specific questions is slightly lower, this difference is also negligible. As mentioned in section 7.8.2.2, this model has the highest score for multiple-choice questions, with 91.91% using Zero-shot prompting and CoT prompting. Both Few-shot and Self-Consistently scored around 10% lower in comparison. Moreover, the trend of new prompting methods improving the outcome of open-ended questions is also present when introducing the filter step, again implying that these types of questions are positively influenced by the extra context or logical structuring offered by these methods. Like the model using RAG MMR with a filter, alternative-response questions do best with Few-shot prompting. This suggests that alternative-response questions might inherently resonate with the precise and targeted context given by Few-shot prompts, especially when the context from RAG is directly related to the question. Conversely, when the context is less accurate, as observed in the baseline model and models without a filter, Few-shot prompting yields the lowest performance across all models for alternative-response questions.

7.8.3 Impact of Prompting Methods

Figure 3 showed that while new prompting methods significantly boosted the Baseline Model, their impact was marginal on the models using RAG. To accurately gauge the effectiveness of various prompting methods when used alongside RAG, all average calculations will omit the baseline model. The most effective prompting method was CoT prompting, achieving an average accuracy of 80.21%, closely followed by Self-consistency at 79.90%. Few-shot and Zero-shot prompting hovered around 78.5%, with Zero-shot being slightly more effective. This pattern was also observed in the responses to general and specific questions but, on average, around 10% lower. The only exception occurred with Few-shot prompting, which slightly outperformed Zero-shot prompt-

ing in specific questions. Nonetheless, these variations in performance were generally small. However, understanding which prompting methods are effective with specific models and question types is vital to selecting the appropriate prompting method based on the given situation.

7.8.3.1 Zero-Shot-Prompting

Although Zero-shot prompting demonstrated the second-lowest overall performance, its effectiveness was still noteworthy. This method particularly excelled in multiple-choice questions with an average accuracy of 84.04%. This is the highest among all other prompting methods. Zero-shot prompting worked especially well in combination with models using cosine similarity, where it achieved nearly 91% accuracy, a point underscored in sections 7.8.2.2 and 7.8.2.4. However, Zero-shot prompting's significant limitation was its performance with open-ended questions. Across all prompting methods, Zero-shot prompting yielded poorer results for open-ended questions, only achieving a score of 76%. This suggests that Zero-shot prompting is less effective at dealing with the complexities of open-ended questions, possibly due to its limited capacity to interpret and adapt to the nuanced context these types of queries often require.

7.8.3.2 Few-Shot-Prompting

Although the margin was small, Few-shot prompting emerged as the least effective method. This trend held across various question types. Despite being competitive in alternative-response and open-ended questions, Few-shot prompting did not outperform other methods in any specific category. Its major drawback was evident in multiple-choice questions, where it achieved only 75% accuracy, the lowest of all. Closer examination reveals that this underperformance in multiple-choice questions was compounded by weaker results in both alternative-response and open-ended questions, particularly in the absence of filters. However, as highlighted in section 7.8.2.4, filters significantly improved Few-shot prompting's effectiveness in these question types. Given this overall weak performance, I suggest reevaluating the results using more examples.

7.8.3.3 Chain of Thought

CoT generally outperformed all other methods. Upon closer examination of different question types, it's apparent that this superior performance is largely attributed to its effectiveness in responding to open-ended questions. CoT achieved a success rate of 82.35%, surpassing all other prompting methods. However, its performance in multiple-choice and alternative-response questions was relatively lower, hovering around 77.5% for both categories. This suggests that CoT's strengths lie more in handling the complexity and nuance of open-ended queries rather than the more structured formats of multiple-choice and alternative-response questions.

7.8.3.4 Self-Consistency

Self-consistency ranked as the second-best category overall, displaying an accuracy nearly on par with that of CoT. Analyzing the performance across different question types reveals a pattern similar to CoT's. Self-consistency showed strong results for open-ended questions, achieving an accuracy of 81.62%. The scores for the other two categories, multiple-choice and alternative-response questions, were also similar, each around 77.5%. Therefore, this indicates that Self-consistency, much like CoT, excels in handling open-ended questions while maintaining consistent, albeit slightly lower, performance in the more structured formats of multiple-choice and alternative-response questions.

7.9 Discussion

7.9.1 Interpretation of Findings

In this research, the efficacy of various models in handling diverse question formats was evaluated through a detailed 60-question survey focused on price personalization. The results highlighted the success of RAG and filtering in significantly enhancing model performance. However, a notable finding from the analysis was that despite this success, all prompting methods showed a diminished

effect in improving the RAG models, starkly contrasting their considerable impact on the baseline model's performance. The RAG models employing MMR and cosine similarity each demonstrated distinct performance profiles. For the RAG models with MMR, a consistent pattern emerged. After introducing new prompting techniques, the model showed enhanced accuracy in open-ended questions but exhibited reduced effectiveness for alternative-response and multiple-choice questions. In contrast, the cosine similarity models were particularly adept at multiple-choice questions. However, while new prompting methods boosted their performance in open-ended questions, they adversely affected their multiple-choice question accuracy. Integrating a filtering step uniformly improved the performance of both MMR and cosine similarity models, particularly in addressing specific, complex queries, with the most notable advancements observed in open-ended questions. Chain of Thought and Self-consistency emerged as the most effective prompting methods, especially for open-ended questions. While not the strongest overall, Few-shot prompting proved particularly efficient for alternative-response questions within the specific framework of RAG models enhanced by filtering. In contrast, though highly effective in multiple-choice questions, Zero-shot prompting demonstrated limitations in handling open-ended questions.

7.9.2 Limitations and Future Work

7.9.2.1 Limitations

Several limitations were evident in the study's methodology and questionnaire design. Using a reduced set of papers for testing might have limited the models' exposure to a wider variety of data, potentially affecting the breadth and depth of the findings. This aspect is particularly relevant since real-world scenarios often predominantly involve open-ended questions, and the current distribution may not fully represent such scenarios. Future studies could benefit from employing a more comprehensive dataset, enhancing the validity and applicability of the results. Additionally, increasing the number of questions in future studies could provide a more comprehensive understanding of the subject matter. Furthermore, the analysis of the baseline model revealed its lower average accuracy compared to other models, particularly in its ability to create reasoning pathways

for specialized queries autonomously. This indicates the need for exploring alternative models, such as the 70B Llama 2 model, which might offer more advanced reasoning capabilities. Lastly, future research efforts should also explore adopting more advanced embedding models and vector stores.

7.9.2.2 Future Work

Looking ahead in the field of language model research, the following areas represent key directions for subsequent studies:

1. **Enhancing Capability for Open-Ended and Specific Questions**: Given the complexity and relevance of real-world applications, prioritizing the enhancement of models' abilities to process specific, detailed open-ended questions is of utmost importance.

2. **Vectorization of Model History**: Implementing advanced vectorization techniques like "VectorStoreRetrieverMemory" (LangChain, Inc., 2023*a*) could significantly enhance the retrieval capabilities of models by offering a more sophisticated method for models to access and utilize their historical data, improving response accuracy and efficiency.

3. **Combining RAG with Classification Tools**: Integrating a classification tool with RAG, such as the "Cohere Reranker" (LangChain, Inc., 2023*b*), could enable better selection of relevant context.

4. **Exploring Advanced Prompting Methods**: Delving into newer prompting methods, such as Tree of Thoughts and Automatic Prompt Engineer, as outlined in the prompt engineering guide (DAIR.AI, 2023), presents an opportunity to explore cutting-edge model interaction.

5. **Employing Fine-Tuning for the Law Domain**: Tailoring models through fine-tuning to suit specific domains, such as legal language, could greatly enhance their applicability and accuracy in specialized fields.

6. **Query Expansion**: Query expansion is a sophisticated technique to augment retrieval by incorporating additional terms into the original query. The benefit of this approach lies in its ability to refine and broaden the search parameters, leading to more accurate and comprehensive search results (Efthimiadis, 1996).

7. **Integration of LLMs with Tools**: Integrating Large Language Models with various tools can significantly enhance their functionality and versatility. This integration enables LLMs to interact with different software platforms and data sources, thereby broadening their scope of application and improving their interactive capabilities (OpenAI, 2023).

8. **Reinforcement Learning with Human Feedback**: RLHF is a powerful method for further aligning LLMs with human preferences and instructions and should be further explored in future research (Touvron et al., 2023).

9. **Evaluation of Model Performance**: Comprehensive evaluation of LLM performance is crucial to understanding their capabilities and limitations accurately. Currently, this topic is still an open research question (Touvron et al., 2023)

Each area offers a pathway to significant advancements in the field, contributing to developing more sophisticated, accurate, and contextually aware language models.

7.10 Conclusion

This study focused on enhancing chatbot performance in the context of price personalization, employing a mix of conventional and innovative methods to evaluate how different models respond to various question types. The findings revealed that while integrating RAG and filtering greatly improves model performance, the effectiveness of different prompting methods varies. Overall, the performance depends on the type of question being asked and the specificity of the information being queried. The study also identifies paths for future research, such as combining RAG with Classification Tools.

8 Group Conclusion

This research has substantiated the practical application of LLMs in the legal domain, specifically focusing on price personalization. The study employed RAG to enhance the capabilities of a Llama 2 model, significantly improving its performance as a chatbot. Furthermore, we demonstrated the model's efficacy in the selection and ranking of legal documents. The synergy between LLMs and RAG presents a considerable leap forward in legal research, enabling the flexible incorporation of external databases. Moreover, we proved that including these databases is instrumental in addressing the limitations inherent in current LLMs, particularly in specialized fields such as price personalization and law.

Moving forward, the study opens several avenues for future research. These include exploring the potential of Query Expansion to refine and broaden the search capabilities of LLMs, further integration of ranking models with chatbots for improved performance, and combining LLMs with various tools to enhance their functionality. In essence, this research establishes a foundation for the continued and expanded use of LLMs in the legal domain, paving the way for more sophisticated, accurate, and contextually aware systems that can adapt to the specific needs of legal research and practice.

References

Ahn, Jaimeen, and Alice Oh. 2021. "Mitigating language-dependent ethnic bias in BERT." *arXiv preprint arXiv:2109.05704*.

Azamfirei, Razvan, Sapna R Kudchadkar, and James Fackler. 2023. "Large language models and the perils of their hallucinations." *Critical Care*, 27(1): 1–2.

Bhattacharya, Paheli, Kaustubh Hiware, Subham Rajgaria, Nilay Pochhi, Kripabandhu Ghosh, and Saptarshi Ghosh. 2019. "A Comparative Study of Summarization Algorithms Applied to Legal Case Judgments." 413–428. Cham:Springer International Publishing.

Blair-Stanek, Andrew, Nils Holzenberger, and Benjamin Van Durme. 2023. "Can GPT-3 perform statutory reasoning?" *arXiv preprint arXiv:2302.06100*.

Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghu-

nathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2022. "On the Opportunities and Risks of Foundation Models."

Brackey, Adrienne. 2019. "Analysis of Racial Bias in Northpointe's COMPAS Algorithm." PhD diss. Tulane University School of Science and Engineering.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. "Language models are few-shot learners." *Advances in neural information processing systems*, 33: 1877–1901.

Cai, Deng, Yan Wang, Lemao Liu, and Shuming Shi. 2022. "Recent advances in retrieval-augmented text generation." 3417–3419.

Carbonell, Jaime, and Jade Goldstein. 1998. "The use of MMR, diversity-based reranking for reordering documents and producing summaries." 335–336.

Chalkidis, Ilias, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Martin Katz, and Nikolaos Aletras. 2022. "LexGLUE: A Benchmark Dataset for Legal Language Understanding in English."

Charlotin, Damien. 2023*a*. "Large Language Models and the Future of Law." *Available at SSRN 4548258*.

Charlotin, Damien. 2023*b*. "Large Language Models and the Future of Law." *Available at SSRN 4548258*.

Chen, Yuh-Shyan, Shin-Wei Chiang, and Meng-Luen Wu. 2022. "A few-shot transfer learning approach using text-label embedding with legal attributes for law article prediction." *Applied Intelligence*, 52(3): 2884–2902.

Cui, Jiaxi, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. 2023. "ChatLaw: Open-Source Legal Large Language Model with Integrated External Knowledge Bases."

DAIR.AI. 2023. "Prompt Engineering Guide." *https://www.promptingguide.ai/*.

Diao, Shizhe, Rui Pan, Hanze Dong, Ka Shun Shum, Jipeng Zhang, Wei Xiong, and Tong Zhang. 2023. "Lmflow: An extensible toolkit for finetuning and inference of large foundation models." *arXiv preprint arXiv:2306.12420*.

Dixon, Lucas, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. "Measuring and mitigating unintended bias in text classification." 67–73.

Do, Phong-Khac, Huy-Tien Nguyen, Chien-Xuan Tran, Minh-Tien Nguyen, and Minh-Le Nguyen. 2017. "Legal Question Answering using Ranking SVM and Deep Convolutional Neural Network."

Efthimiadis, Efthimis N. 1996. "Query Expansion." *Annual review of information science and technology (ARIST)*, 31: 121–87.

Fang, Yin, Xin Tian, Hao Wu, Songyuan Gu, Zhu Wang, Feng Wang, Junliang Li, and Yang Weng. 2020. "Few-shot learning for Chinese legal controversial issues classification." *IEEE Access*, 8: 75022–75034.

Felkner, Virginia K, Ho-Chun Herbert Chang, Eugene Jang, and Jonathan May. 2022. "Towards WinoQueer: Developing a benchmark for anti-queer bias in large language models." *arXiv preprint arXiv:2206.11484*.

Han, Kai, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing XU, and Yunhe Wang. 2021. "Transformer in Transformer." Vol. 34, 15908–15919. Curran Associates, Inc.

Hemmatian, Babak, Razan Baltaji, and Lav R. Varshney. 2023. "Muslim-Violence Bias Persists in Debiased GPT Models."

Hou, Xinyi, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu Wang. 2023. "Large Language Models for Software Engineering: A Systematic Literature Review."

Huang, Quzhe, Mingxu Tao, Chen Zhang, Zhenwei An, Cong Jiang, Zhibin Chen, Zirui Wu, and Yansong Feng. 2023. "Lawyer LLaMA Technical Report."

Jain, Deepali, Malaya Dutta Borah, and Anupam Biswas. 2021. "Summarization of legal documents: Where are we now and the way forward." *Computer Science Review*, 40: 100388.

Katz, Daniel Martin, Dirk Hartung, Lauritz Gerlach, Abhik Jana, and Michael J. Bommarito II au2. 2023. "Natural Language Processing in the Legal Domain."

Kojima, Takeshi, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. "Large language models are zero-shot reasoners." *Advances in neural information processing systems*, 35: 22199–22213.

Lai, Jinqi, Wensheng Gan, Jiayang Wu, Zhenlian Qi, and Philip S. Yu. 2023. "Large Language Models in Law: A Survey."

LangChain, Inc. 2023*a*. "Backed by a Vector Store." *https://python.langchain.com/docs/modules/memory/types/vectorstore_retriever_memory*.

LangChain, Inc. 2023*b*. "Cohere Reranker." *https://python.langchain.com/docs/integrations/retrievers/cohere-reranker?ref=blog.langchain.dev*.

LangChain, Inc. 2023*c*. "Long Context Reorder." *https://python.langchain.com/docs/modules/data_connection/document_transformers/post_retrieval/long_context_reorder*, Accessed: 2023-12-18.

LangChain, Inc. 2023*d*. "Question Answering with LangChain." *https://python.langchain.com/docs/use_cases/question_answering/*, Accessed: 2023-12-18.

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. "Retrieval-augmented generation for knowledge-intensive nlp tasks." *Advances in Neural Information Processing Systems*, 33: 9459–9474.

Li, Haitao, Changyue Wang, Weihang Su, Yueyue Wu, Qingyao Ai, and Yiqun Liu. 2023*a*. "THUIR@COLIEE 2023: More Parameters and Legal Knowledge for Legal Case Entailment."

Liu, Nelson F., Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023*a*. "Lost in the Middle: How Language Models Use Long Contexts."

Liu, Zhengzhong, Aurick Qiao, Willie Neiswanger, Hongyi Wang, Bowen Tan, Tianhua Tao, Junbo Li, Yuqi Wang, Suqi Sun, Omkar Pangarkar, et al. 2023*b*. "LLM360: Towards Fully Transparent Open-Source LLMs." *arXiv preprint arXiv:2312.06550*.

Li, Yingji, Mengnan Du, Rui Song, Xin Wang, and Ying Wang. 2023*b*. "A Survey on Fairness in Large Language Models."

Meta AI. 2023. "Llama Model Downloads." *https://ai.meta.com/resources/models-and-libraries/llama-downloads/*.

Naveed, Humza, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. "A Comprehensive Overview of Large Language Models."

Nawar, Afra, Mohammed Rakib, Salma Abdul Hai, and Sanaulla Haq. 2022. "An Open Source Contractual Language Understanding Application Using Machine Learning." 42–50. Marseille, France:European Language Resources Association.

Niklaus, Joel, and Daniele Giofré. 2022. "BudgetLongformer: Can we Cheaply Pretrain a SotA Legal Language Model From Scratch?"

OpenAI. 2023. "Applying OpenAI's RAG Strategies." *https://blog.langchain.dev/applying-openai-rag/.*

Paaß, Gerhard, and Sven Giesselbach. 2023. "Knowledge Acquired by Foundation Models." *Foundation Models for Natural Language Processing: Pre-trained Language Models Integrating Media*, 161–185. Cham:Springer International Publishing.

Panda, Swetasudha, Ari Kobren, Michael Wick, and Qinlan Shen. 2022. "Don't Just Clean It, Proxy Clean It: Mitigating Bias by Proxy in Pre-Trained Models." 5073–5085.

Paul, Shounak, Arpan Mandal, Pawan Goyal, and Saptarshi Ghosh. 2023. "Pre-trained Language Models for the Legal Domain: A Case Study on Indian Law."

Rabelo, Juliano, mi-young Kim, and Randy Goebel. 2021. "The Application of Text Entailment Techniques in COLIEE 2020." 240–253.

Rahutomo, Faisal, Teruaki Kitasuka, and Masayoshi Aritsugi. 2012. "Semantic cosine similarity." Vol. 4, 1.

Rosa, Guilherme Moraes, Luiz Bonifacio, Vitor Jeronymo, Hugo Abonizio, Roberto Lotufo, and Rodrigo Nogueira. 2022. "Billions of parameters are worth more than in-domain training data: A case study in the legal case entailment task." *arXiv preprint arXiv:2205.15172*.

Stahl, Maja, Maximilian Spliethöver, and Henning Wachsmuth. 2022. "To prefer or to choose? generating agency and power counterfactuals jointly for gender bias mitigation." 39–51.

Strickson, Benjamin, and Beatriz De La Iglesia. 2020. "Legal Judgement Prediction for UK Courts." *ICISS '20*, 204–209. New York, NY, USA:Association for Computing Machinery.

Sun, Zhongxiang. 2023. "A Short Survey of Viewing Large Language Models in Legal Aspect."

Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas

Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. "Llama 2: Open Foundation and Fine-Tuned Chat Models."

Utils, GPU. 2023. "Llama 2 Prompt Template." *https://gpus.llm-utils.org/llama-2-prompt-template/.*

Wang, Xuezhi, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. "Self-consistency improves chain of thought reasoning in language models." *arXiv preprint arXiv:2203.11171*.

Webster, Kellie, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. "Measuring and reducing gendered correlations in pre-trained models." *arXiv preprint arXiv:2010.06032*.

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. "Chain-of-thought prompting elicits reasoning in large language models." *Advances in Neural Information Processing Systems*, 35: 24824–24837.

Wu, Yiquan, Kun Kuang, Yating Zhang, Xiaozhong Liu, Changlong Sun, Jun Xiao, Yueting Zhuang, Luo Si, and Fei Wu. 2020. "De-Biased Court's View Generation with Causality." 763–780. Online:Association for Computational Linguistics.

Wu, Yiquan, Siying Zhou, Yifei Liu, Weiming Lu, Xiaozhong Liu, Yating Zhang, Changlong Sun, Fei Wu, and Kun Kuang. 2023. "Precedent-Enhanced Legal Judgment Prediction with LLM and Domain-Model Collaboration."

Xiao, Chaojun, Xueyu Hu, Zhiyuan Liu, Cunchao Tu, and Maosong Sun. 2021. "Lawformer: A pre-trained language model for Chinese legal long documents." *AI Open*, 2: 79–84.

Yan, Eugene. 2023. "Open LLMs." *https://github.com/eugeneyan/open-llms*.

Ye, Hai, Xin Jiang, Zhunchen Luo, and Wenhan Chao. 2018. "Interpretable Charge Predictions for Criminal Cases: Learning to Generate Court Views from Fact Descriptions."

Zhao, Wayne Xin, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. "A Survey of Large Language Models."

Zhong, Haoxi, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. "How Does NLP Benefit Legal System: A Summary of Legal Artificial Intelligence." 5218–5230. Online:Association for Computational Linguistics.

Zhou, Yongchao, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. "Large Language Models Are Human-Level Prompt Engineers."

Zmigrod, Ran, Sabrina J Mielke, Hanna Wallach, and Ryan Cotterell. 2019. "Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology." *arXiv preprint arXiv:1906.04571*.

9 Appendix

To gain access to the underlying code repositories, request access by contacting Lorenzo Schumann (56178@noavsbe.pt) or Anton Badort (55358@novasbe.pt).

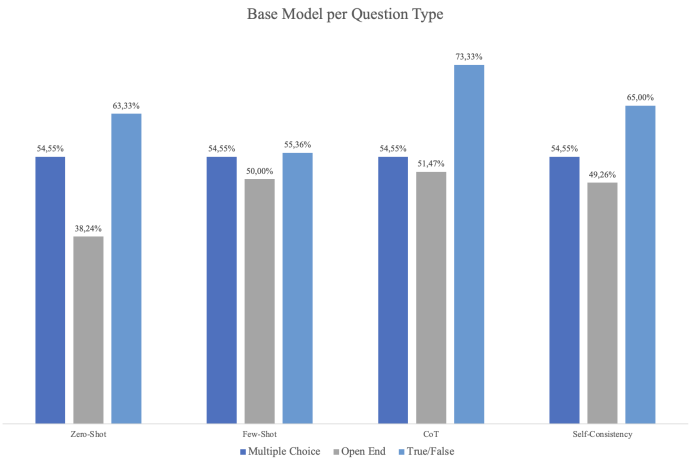

Figure 6: Base Model per Question Type

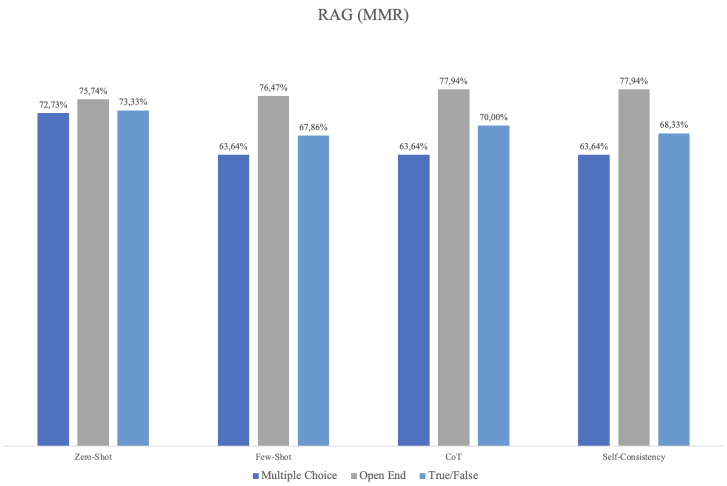

Figure 7: RAG (MMR) per Question Type

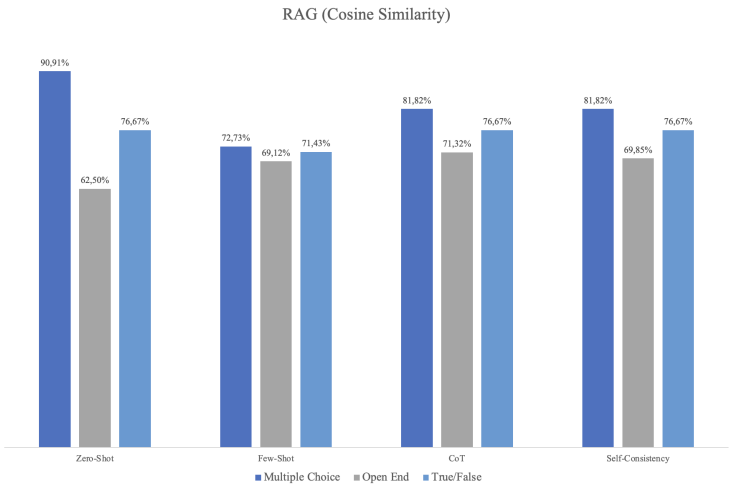

Figure 8: RAG (Cosine Similarity) per Question Type

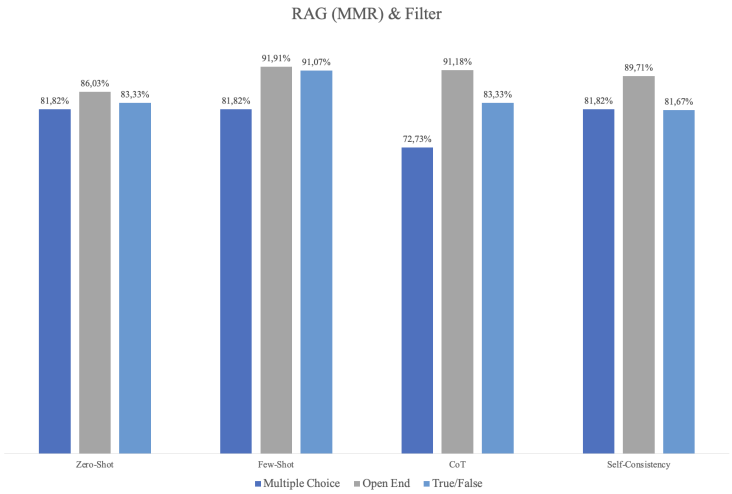

Figure 9: RAG (MMR) and Filter) per Question Type

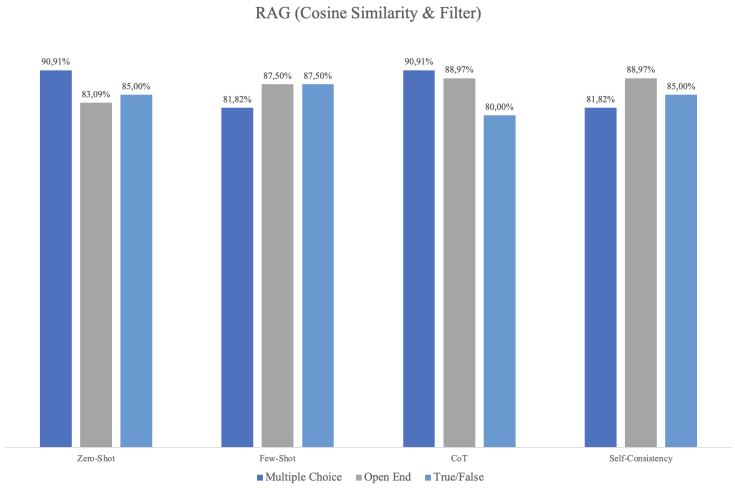

Figure 10: RAG (Cosine Similarity) and Filter) per Question Type

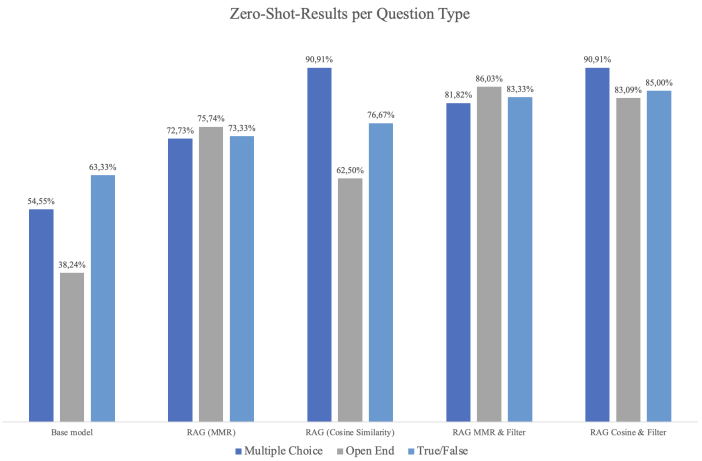

Figure 11: Few-shot per Question Type

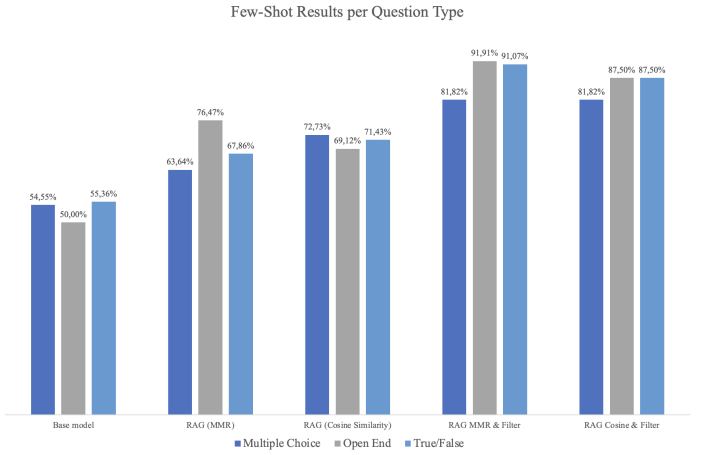

Figure 12: Few-shot per Question Type

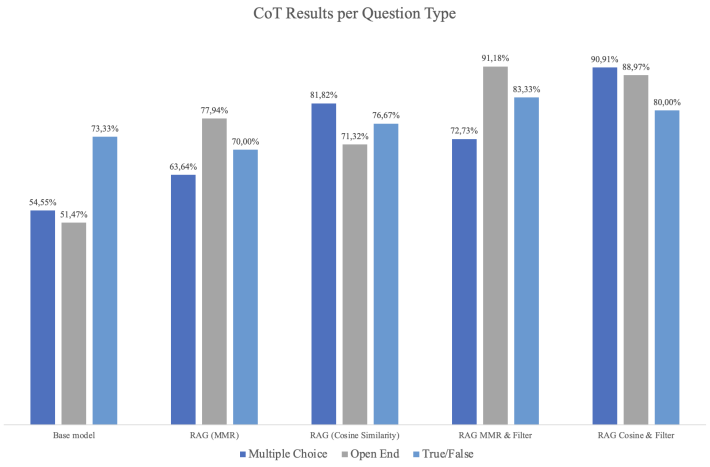

Figure 13: CoT per Question Type

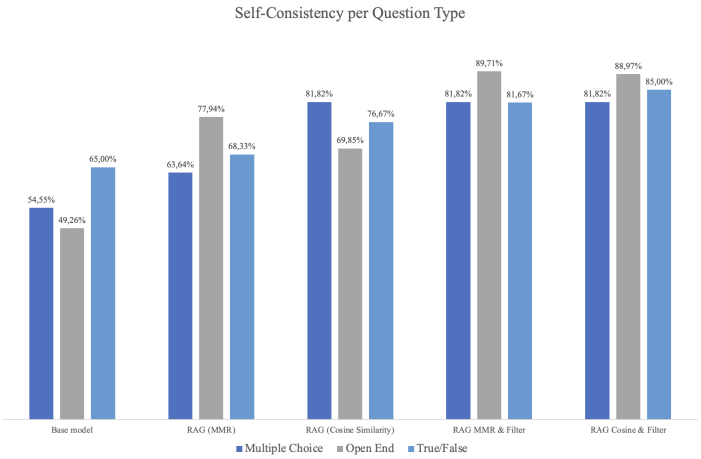

Figure 14: Self-consistency per Question Type