Integrating LLMs and Chatbots Technologies - A Case Study on Brazilian Transit Law

[1]Rafael Selau M. Rocho,[1]Anderson Luiz Fernandes Perez,
[2]Giovani P. Farias, and[1]Alison R. Panisson

[1] Department of Computing (DEC)
Federal University of Santa Catarina (UFSC), Araranguá, Brazil
alison.panisson@ufsc.br
[2] Center for Computational Sciences (C3)
Federal University of Rio Grande (FURG), Rio Grande, Brazil
giovanifarias@gmail.com

Abstract. This paper proposes a hybrid integration of technologies for the development of chatbots. The proposed integration incorporates a framework for developing rule-based chatbots and Large Language Models (LLMs). As a case study of the proposed integration, an application was developed focused on Brazilian traffic legislation, specifically Law No. 9.503 of Sept. 23, 1997, Chapter XV - Infractions. The study demonstrates the technical feasibility of the proposed integration and addresses related challenges. It also identifies future research opportunities, such as adapting the chatbot to different laws and enhancing accessibility. In general, the study shows the potential of combining chatbot development frameworks with LLM to create sophisticated personalized chatbots.

1 Introduction

In recent years, artificial intelligence has simplified processes and enhanced interactions between computer systems and humans, particularly through the development of conversational agent-autonomous programs designed for interactive conversations, such as Google's Gemini[3] and OpenAI's ChatGPT[4].

The main objective of this work is to explore the integration between rule-based chatbot technologies, such as Rasa, which is highly customizable and versatile chatbot development platform, and Large Language Models (LLMs) using the OpenAI API. This proposal aligns with recent studies that highlight the use of LLMs as a complementary tool, enhancing the quality of responses in chatbot applications [13]. The proposed integration represents a significant advancement in the ability to create sophisticated and intelligent chatbots, highlighting the opportunities and challenges that this integration offers.

To evaluate the integration proposed in this work, a chatbot was developed to answer questions related to Brazilian traffic legislation, specifically Law No.

[3] https://gemini.google.com/
[4] https://chatgpt.com/

9.503 [3], with a primary focus on Chapter XV - Infractions. Traffic violations can result in penalties such as fines, suspension of driving privileges, and legal action. In addition, traffic accidents are a major public health problem in Brazil, causing about 45 thousand deaths annually, with the country ranked among the highest in traffic related deaths worldwide [1, 8]. The complexity of these laws and lack of awareness often contribute to these high accident rates.

This work explores the benefits and challenges of integrating Rasa with the OpenAI API and implementing a chatbot for Law No. 9.503/1997, Chapter XV - Infractions. The chatbot serves as a case study to show the potential of this integration for AI solutions, highlighting its applicability to Brazilian traffic laws.

2 Background

2.1 Chatbot Technologies

Chatbots play a crucial role in the field of artificial intelligence and are designed to simulate human interaction [2]. They have been increasingly used in a variety of sectors, including customer service, technical support, marketing, and e-Commerce [16]. The development of chatbots has significantly grown in recent years, driven by the increased adoption of artificial intelligence technologies. According to a report by Grand View Research [14], the growing customer demand for self-service operations has fueled the need to provide customer services 24/7, while also seeking to reduce operational costs by delegating tasks to chatbots. This rise in customer demand for self-service has been a driving factor in the increased adoption of such technologies.

2.2 Large Language Models

Large Language Models (LLMs) are a class of AI models capable of understanding and generating text in a way that closely resembles human language. They are trained on large data sets to learn patterns of language, grammar, semantics, and even cultural context. As the parameter size of LLMs continues to grow, recent studies have indicated that LLMs can develop remarkable capabilities [5]. Examples such as OpenAI's GPT-3.5 and GPT-4, Anthropic's Claude, and Google's Gemini are noted for their ability to perform a wide range of natural language processing tasks.

LLMs consist of deep neural networks that can take text input and generate output that is coherent and relevant to the input provided. According to [6], recent studies have shown that LLMs exhibit impressive generalization and reasoning abilities, allowing them to generalize better across various unseen tasks and domains. Instead of requiring fine-tuning for each specific task, LLMs can apply their learned knowledge and reasoning skills to new tasks simply by providing appropriate instructions or a few task demonstrations. This makes these models incredibly versatile for tasks such as [10]: (a) **Text Generation**: One of the most notable uses of LLMs is the generation of coherent and natural text.

They can create stories, poetry, programming code, and much more based on specific inputs; (b) **Natural Language Processing (NLP) Tasks**: Beyond text generation, LLMs are used in a variety of NLP tasks, such as machine translation, question answering, text summarization, etc.; and (c) **Practical Applications**: LLMs have a wide range of real-world applications. They can be used in chatbots, virtual assistants, text editors, translations, intelligent search, automatic text generation, and more.

Despite being effective tools in natural language processing, LLMs also face several challenges and difficulties, such as [9]: (a) **Coherence and Context**: LLMs can generate responses that seem plausible but may lack long-term cohesion and context. Maintaining coherent and relevant conversations is particularly challenging in longer interactions; (b) **Ambiguity**: Language models often struggle with ambiguity in language, which can lead to incorrect interpretations or ambiguous responses; (c) **Limited Knowledge**: LLMs have a knowledge cut-off date, which means that they only contain information available up to a certain point when they were trained. (d) **Privacy and Security Challenges**: Indiscriminate use of LLMs for content generation can create security issues, such as the spread of misinformation and privacy threats; and (e) **Robustness**: Language models are sensitive to small changes in input and may produce inconsistent or incorrect responses under minimal variations in context.

2.3 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) is an innovative and efficient approach that improves the quality of responses generated by LLMs by grounding the model in external sources of knowledge to supplement the LLM's internal information representation [21]. LLMs can be inconsistent, sometimes providing random facts from their training data. They cannot process an entire knowledge base at once due to text length limits and high data transmission costs. According to [12], LLMs perform better when relevant information is placed at the beginning or end of the input, with performance declining as the length of the context increases, even in models designed for longer contexts.

As a solution, RAG methods [11] offer a promising approach for LLMs to interact effectively. At its core, RAG seeks to enhance text generation by incorporating information retrieval mechanisms. It operates with two main components: the language generation model, which creates coherent and meaningful text, and an information retrieval model, which incorporates the ability to retrieve specific information from external sources [21]. RAG represents an exciting frontier in artificial intelligence, promising significant advances in the generation of high-quality, data-enriched, and contextually relevant text. This approach has the potential to transform how we interact with information, creating more accurate and informative content in various contexts, from virtual assistants to content generation for research and dissemination of knowledge.

3 Integrating Chatbots and LLMs

As pointed out in the literature, a hybrid approach combining rule-based chatbots and LLMs has shown great promise in developing sophisticated chatbots that meet the needs of specific domains [13]. In this context, we propose an integration of chatbots developed with the Rasa framework and LLMs, particularly using the RAG approach.

The implementation of RAG in an LLM-based question-answering system ensures that the model has access to the most recent and reliable information. This characteristic is especially important in an application domain in which the subject constant changes. By integrating an LLM into a set of external and verifiable facts, the model has fewer opportunities to rely on the information embedded in its parameters. This reduces the chances of a LLM responding with incorrect or misleading information [21].

An overview of the RAG-based approach can be seen in Figure 1. The process begins with a user input, for example, asking "How does it work?" A search process is then triggered (1), focusing on finding the user's query and locating the most relevant content in the knowledge base to answer it (2). Once relevant data is extracted from the knowledge base, it is sent along with the user's question to the LLM (3). The LLM then assimilates the provided information and provides the answer to the question.

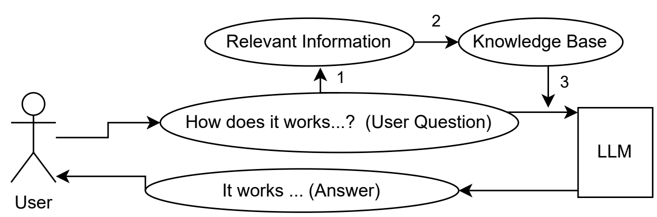

Fig. 1. RAG Interaction Diagram.

Chatbots have become an essential part of online interactions, whether to improve customer experience, automate tasks, or provide support. Integrating Rasa, a popular chatbot development framework and one of the most widely used in the world [15], with OpenAI's LLM through its API, a powerful natural language processing tool, facilitates the creation of intelligent and versatile chatbots. The development steps of the proposed integration are outlined below:

1. Planning and Design: The first step is to define the purpose and objectives of the chatbot. What problem will it solve? What questions will it answer? How will it fit into the business strategy? Additionally, it is important to design the chatbot architecture, including conversation flows, user intentions, and chatbot responses. The following steps will explore the development process of this chatbot, along with the tools used.
2. Rasa Model Training: Rasa uses machine learning to understand user intentions and generate appropriate responses. The language model is trained with training data, including examples of dialogues and user intentions. Rasa

provides a range of tools to train, evaluate, and adjust the model until it achieves satisfactory performance.
3. Integration with OpenAI API: The OpenAI API is used as an interface to an LLM, enriching the chatbot's ability to process natural language and generate more contextually relevant and human-like responses. While OpenAI API was used in this work, other LLMs can be easily incorporated without great changes at the proposed architecture.
4. Prompt Engineering: A prompt is defined to add instructions to the LLM, using the RAG approach described, with the goal of achieving domain specific and more satisfactory responses.

3.1 ChatBot Development with Rasa

The decision to use Rasa was based on its open-source structure to develop AI assistants and chatbots, as well as its widespread use. Rasa offers a versatile platform that allows developers to create highly customized chatbots capable of handling specific business logic of the application being developed. One of Rasa's key features is its ability to intelligently handle user interaction, based on its Natural Language Understanding (NLU). Rasa's NLU is trained to identify user intents and extract relevant elements from messages. This enables the chatbot to understand user questions and requests, providing more accurate and relevant responses. For example, below is a snippet of a dataset used to train the NLU, with sentences used to classify the user's greeting intention:

```
- intent: greeting
    examples: |
      - Hello
      - Good afternoon!
      _ Good morning!
      - Hi!
```

It is also possible to extract relevant information from user-interacted sentences according the application need, for instance, extracting the user's name through an entity `name` after the chatbot requests this information through the interaction "`To proceed, I need to know your name. What is your name?`", following with the intent `provide_name`:

```
- intent: provide_name
    examples: |
      - My name is [Jonas](name)
      - [Maria](name)
      - Ah yes, my name is [Thiago](name)
      - Yes, it's [Alex](name)
```

After the NLU recognizes the user's intent and associated entities from the received interaction, Rasa core processes this information, defining the conversation flow, and selecting the chatbot's actions in response to that user interaction.

Chatbot responses are encoded in the form of actions, which can be simple, i.e. a direct mapping to a list of texts that can be offered as a response, or they can be implemented as customizable methods by the chatbot developer, implemented in the Python programming language, offering all the resources of external libraries, integration to APIs, etc.

Two custom actions, `ActionGetKeywords` and `ActionAnswerQuestion`, were implemented. These custom actions are used to implement the RAG method, i.e., to assemble the prompt used for requests to the OpenAI API. The action `ActionGetKeywords` is used to obtain relevant keywords from the user's interaction, while `ActionAnswerQuestion` is responsible for making the request to the OpenAI API and formulating responses based on the available information.

The integration of the OpenAI API with Rasa is a key element of this work. While Rasa handles user interaction and collects relevant entities from the user's interaction, through the RAG method for prompt generation, the OpenAI API plays an important role by generating responses based on the available information. This allows the chatbot assistant to provide higher-quality responses, enriching the user experience.

The use of Rasa integrated with the OpenAI API represents an efficient approach to creating intelligent chatbots and highly customized AI assistants. Rasa handles natural language understanding and collects relevant information, RAG prompt engineering retrieves the relevant information to formulate correct answers, and the OpenAI API formulates the answer and provides more humanized responses. This combination offers great potential, automating tasks and providing effective personalized assistance.

3.2 Prompt Engineering

OpenAI API is a tool that allows interaction with advanced language models, such as GPT-3.5, to generate intelligent responses from customized prompts. Prompt engineering plays a crucial role in harnessing this technology, as it helps guide the model in generating relevant and accurate responses.

To integrate Rasa and the API, specific actions were developed within the Rasa framework to perform prompt assembly through the described RAG method, as well as to query the API and assemble the user response. The first of the actions implemented was the action to obtain keywords. The *ActionGetKeywords* class defines a custom action that extracts keywords from a question asked by the user. The question is passed as a prompt to the OpenAI API, which returns a list of relevant keywords. These keywords are then stored in a slot called "keywords". Next, the action to answer the question was implemented. The *ActionAnswerQuestion* class defines another custom action that uses the extracted keyword, obtaining relevant information from the legal document to build a prompt. The question, intention, entities, and domain information are then used to build a final prompt to the OpenAI API, which returns an answer.

The designed system uses the OpenAI API to perform different tasks using pre-defined prompt models that include instructions to guide the LLM in reaching the final answer and avoid unwanted or incorrect responses. Along with this, it queries information from the Brazilian Traffic Code - Law No. 9.503 [3], and provide that information to the API extracts the data, and answer the user's question. The goal of the proposed prompt engineering was to correct and avoid responses that were being generated in another language, responses that were not in line with the question or were being misinterpreted by the LLM.

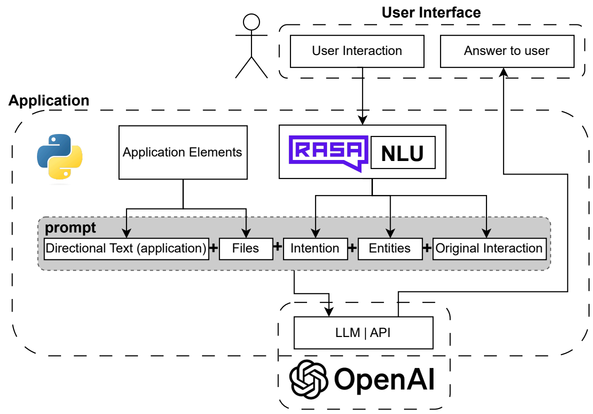

Fig. 2. Prompt Engineering.

The prompt engineering also guides the LLM to generate (i) responses only in the language of the application; (ii) objective responses; and (iii) synonyms of keywords; among others. An illustration of the strategy used in our approach for prompt engineering can be seen in Figure 2.

4 A Case Study on Brazilian Transit Law

4.1 Brazilian Transit Law - The Structure of a Brazilian Legal Document and Its Challenges

The structure of legal documents in Brazil follows a standard established by national legislation [4]. The main components are as follows:

- **Chapters**: Laws are often divided into chapters, organizing the content into sections. Each chapter is identified by a Roman numeral (for example, I, II, III, etc.) and a corresponding title. In this work, Chapter XV - Infractions is the primary focus.
- **Articles**: They are the main units of the legislated material. Articles are numbered ordinally from the first to the ninth (art. 1º, art. 2º, ..., art. 9º) and cardinally from the tenth onward (art. 10, art. 11, ...).
- **Paragraphs**: They are subdivisions of articles that may contain explanations or modifications of the preceding proposition. Paragraphs are denoted by the symbol “§” followed by a number (for example, § 4º, § 15, etc.).
- **Caput**: Derived from the Latin term for “head,” this refers to the main part of an article, especially when it includes sections and/or paragraphs. This initial part represents the fundamental statement of the article.
- **Sections**: They are generally used to describe situations, hypotheses, or listings related to the rule established at the beginning of the article (caput). In legal documents writing, they are represented by Roman numerals (for example, I, II, III, etc.).
- **Subsections**: They list and detail situations, hypotheses, exceptions, and other elements. Subsections are subdivisions of sections and are represented by lowercase letters (for example, a, b, c, etc.).

One of the main challenges of working with laws is the linguistic complexity that involves specific technical and legal terms, as well as constant updates and changes in legislation, including the approval of new laws, decrees, and constitutional amendments, among others.

4.2 Evaluation

In this Section, we evaluate the proposed approach towards a case study based on the Brazilian Traffic Code (CTB) domain. To evaluate the proposed integration, we have asked diverse questions to the chatbot containing the final integration, evaluating if the answer contains correct information.

Scenario 1: In the first scenario, the user queries the chatbot regarding the severity associated with driving without a seatbelt. The excerpt below presents some examples provided to train Rasa's NLU to recognize this user intention:

```
- intent: seatbelt_severity
  examples:
    - What is the severity of driving without a seatbelt?
    - What are the penalties for not wearing a seatbelt?
    - Tell me what the punishment is for not using the seatbelt?
    ...
    - What is the punishment for not wearing a seatbelt?
    - What is the fine for driving without a seatbelt?
```

In this scenario, the first interaction we evaluated is presented below, in which the user engages with the chatbot with the input, to which the chatbot promptly responds as follows:

```
Your input -> What is the severity of driving without a seatbelt?
Answer -> The severity of driving without a seatbelt is serious.
The penalty is a fine. An administrative measure is the vehicle retention
until the seatbelt is fastened by the offender.
```

The answer is correct and precise, providing only the information necessary to answer the input question. After, following the same user's intention, the user also asks a second question focusing on the penalty aspect of driving without a seatbelt, obtaining the following correct result:

```
Your input -> Tell me what the penalty would be for not using the seatbelt?
Answer -> The penalty for not using the seatbelt is severe. The fine is
R$195.23 and 5 points on the driver's license. The vehicle will be retained
until the seatbelt is fastened by the offender.
```

The answer is correct and precise, providing all relevant information related to the question and avoiding an extensive answer with irrelevant information. Next, we evaluated a scenario where the user's interaction remained focused on the intention related to the severity of not using the seatbelt. However, the user asked a question that contained spelling errors. Even so, the chatbot was able to properly understand and answer the question, which we argue is resulting from the proposed approach, in particular the RAG prompt methodology.

Scenario 2: In the second scenario, the user performs interaction related to stopping the car on the safety line. Below we exemplify part of the NLU training examples used to this specific question, with examples of interactions that recognize this intention from the user:

```
- intent: stop_line_severity
  examples: |
    - What is the severity of stopping on the safety line?
    - What is the severity of stopping on the pedestrian crossing line?
    - What is the fine for stopping the car on the safety line?
    ...
    - What is the punishment for leaving the car on the pedestrian crossing line?
    - What happens if I leave the car on the pedestrian crossing line?
```

As in previous scenarios, a user interacts with the chatbot to evaluate its answers. Below is an example of an interaction carried out with the chatbot, in the context of the above intention, and the response presented by the chatbot:

```
Your input -> I am driving and end up stopping the car on top of the
pedestrian crossing, is there any punishment for this?
Answer -> In other words, you are stopping your car at the pedestrian
crossing, which is a medium infraction. The punishment for this is a
fine of R$130.16 and 4 points on the driver's license. It is important
to follow traffic rules to ensure everyone's safety.
```

The answer is correct and complete, and the chatbot plays a humanized role by providing some advice. This humanized response results from the interaction input, where the proposed integration also considers the user's input to build the prompt, allowing the LLM to answer according to the user's interaction style.
Scenario 3: In this scenario, the user asks the chatbot about car races. Below, we show part of the NLU training file, containing examples of interaction to recognize this particular intention.

```
- intent: race_severity
  examples: |
    - What is the severity of racing?
    - What are the penalties for racing?
    - What is the fine for racing on the road?
    ...
    - What is the punishment for racing on the road?
    - What happens if I race on the road?
```

In the interaction below, we show a scenario in which the user asks about being allowed to race cars, followed by the chatbot's response:

```
Your input -> Can I race on the road?
Answer -> You can race on the road, but it is important to remember that this is
illegal and can have serious consequences. The infraction is considered
"very serious" and can result in a fine of 10 times the maximum amount, suspension
of driving rights, and vehicle seizure. Additionally, the administrative measure of
confiscating the driver's license and vehicle removal may occur. In case of
recurrence within 12 months, the fine will be doubled.
```

The answer is correct, although it is not very direct. The chatbot starts by saying "You can do it, but it is illegal." While this is a common pattern in human dialogues and is incorporated into LLMs through their training, the chatbot could be more straightforward. Note that this interaction also results from how the user interacts with the chatbot.
Scenario 4: In this scenario, the user's intention is to obtain information related to running a red light. Below we present part the NLU training file with examples to recognize this intention:

```
- intent: red_light_severity
  examples: |
    - What is the severity of running a red light?
    - What are the penalties for running a red light?
    - If I run the red light?
    ...
    - What is the punishment for running a red light?
    - Fine for running a red light?
```

In this scenario, the user asks the chatbot about the fine for running a red light, and the chatbot answers as follows:

```
Your input -> What is the fine for running a red light?
Answer -> The fine for running a red light is very serious,
with a fine of R$293.
```

The answer is correct and very precise, providing not only the monetary value but also the fine classification. In summary, the evaluation in the CTB domain highlights the effectiveness of the chatbot in understanding a variety of user interactions, even in the face of challenges such as spelling errors in some cases. Overall, the chatbot demonstrated consistency in providing accurate and contextually appropriate responses, capable of retrieving the correct information from the CTB and crafting a highly humanized response.

5 Related Work

Some works apply similar technologies to implement sophisticated chatbot applications. For example, in [19], the authors propose a conversational dialogue system for nutritional queries that integrates tools from Rasa, ChatGPT, and the Edamam API[5] (a platform that provides nutrition-related information). According to the author, the main goal was to create a conversational assistant that offers a flexible, modular approach capable of providing personalized recommendations, handling dialogues, and offering contextually aware responses.

In [17], the authors introduce a conversational agent integrating Rasa, the GPT-3.5-turbo model from OpenAI, and the Docker-Compose tool (a tool that facilitates the definition, configuration, and execution of multi-container applications in environments). The work aims to explore specific tasks of Natural Language Generation (NLG). According to the author, the work delivers a conversational agent to support learning for Computer Engineering and Mathematics courses at the University of Barcelona, providing assistance to students with questions about the curriculum, subjects, general concepts, and also providing exercises for students to solve. Also, in [7], the authors introduce an approach for a conversational agent to create trigger-action rules and control smart objects in smart environments, such as a smart home. The work utilizes the integration of ChatGPT for dialogue generation and Rasa for handling intents and entities.

To the best of our knowledge, our work is one of the first to deal with legal documents using the RAG approach to retrieve relevant information from

[5] https://www.edamam.com/

these complex structured documents. Additionally, our work is the first to apply chatbot technologies to answer questions about the Brazilian Traffic Code - Infractions, although the proposed application was used solely to validate the proposed technological integration.

6 Conclusion

In conclusion, this work underscores the growing importance of integrating chatbot and LLM technologies to create sophisticated solutions for human-computer interaction. The chatbot developed demonstrates significant potential to improve user interaction by offering more accurate and human-like responses, while also providing alternatives to develop technologies that support various businesses.

In the proposed approach, a rule-based chatbot handles natural language understanding, identifying user intentions and extracting relevant information, while the LLM generates more humanized and informative responses. This combination enhances user interaction and can be used to automate tasks and provide personalized assistance effectively.

The study also proposed an application focused on Brazilian legislation, specifically CTB - Law No. 9,503/1997. Despite the challenges posed by complex legal documents, the chatbot could serve as an informative tool capable of educating drivers and pedestrians about traffic laws. Given the high rates of traffic-related injuries and fatalities, a chatbot that clarifies traffic violations or other legal matters could be crucial in spreading knowledge of Brazilian laws.

However, developing a chatbot, like the one proposed, presents several challenges, including integrating technologies, training natural language models, and ensuring accurate information delivery. Continuous maintenance and updates are crucial to reflect changes in legislation. The chatbot must be precise and informative, offering answers aligned with current laws, highlighting how technology can simplify access to legal information and disseminate knowledge.

In summary, integrating chatbots with LLMs to create a chatbot focused on Brazilian traffic laws exemplifies how technology can simplify human-computer interaction. This approach paves the way for developing effective chatbots in various fields and showcases the potential of AI to streamline processes and enhance user interaction. Ongoing exploration and improvement of these technologies by developers and researchers are essential for benefiting society as a whole. In future work, we intend to explore more sophisticated interaction interfaces, such as chabots able to argue with users [18, 20].

References

1. Abreu, D.R.d.O.M., Souza, E.M.d., Mathias, T.A.d.F.: Impacto do código de trânsito brasileiro e da lei seca na mortalidade por acidentes de trânsito. Cadernos de Saúde Pública **34**(8), (2018).
2. Bansal, H., Khan, R.: A review paper on human computer interaction. International Journal of Advanced Research in Computer Science and Software Engineering **8**(53) (2018)

3. BRASIL: Lei nº 9.503, de 23 de setembro de 1997. (1997), http://www.planalto.gov.br/ccivil_03/leis/L9503.htm
4. BRASIL: Lei complementar nº 95, de 26 de fevereiro de 1998 (1998), https://www.planalto.gov.br/ccivil_03/leis/lcp/lcp95.htm
5. Chen, Z., Mao, H., Li, H., Jin, W., Wen, H., Wei, X., Wang, S., Yin, D., Fan, W., Liu, H., et al.: Exploring the potential of large language models (llms) in learning on graphs. arXiv:2307.03393 (2023)
6. Fan, W., Zhao, Z., Li, J., Liu, Y., Mei, X., Wang, Y., Tang, J., Li, Q.: Recommender systems in the era of large language models (LLMs). arXiv:2307.02046 (2023)
7. Gallo, S., Malizia, A., Paternò, F.: Towards a chatbot for creating trigger-action rules based on chatgpt and rasa. Symposium on End-User Development (2023)
8. Henrique, C.R.d.C.: Custos dos acidentes de trânsito no brasil: Estimativa simplificada com base na atualização das pesquisas do ipea sobre custos de acidentes nos aglomerados urbanos e rodovias. Instituto de Pesquisa Econômica Aplicada (2020)
9. Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., McHardy, R.: Challenges and applications of large language models. arXiv:2307.10169 (2023)
10. Kasneci, E., Seßler, K., Küchemann, et al.: Chatgpt for good? on opportunities and challenges of large language models for education. Learning and individual differences **103**, 102274 (2023)
11. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems **33**, 9459–9474 (2020)
12. Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P.: Lost in the middle: How language models use long contexts. arXiv:2307.03172 (2023)
13. McTear, M., Varghese Marokkie, S., Bi, Y.: A comparative study of chatbot response generation: Traditional approaches versus large language models. In: International Conference on Knowledge Science, Engineering and Management. pp. 70–79. Springer (2023)
14. Research, G.V.: Chatbot market size, share & trends, analysis report by application (customer services, branding & advertising), by type, by vertical, by region (north america, europe, asia pacific, south america), and segment forecasts, 2023 - 2030 (2021), https://www.grandviewresearch.com/industry-analysis/chatbot-market
15. Sharma, R.K., Joshi, M.: An analytical study and review of open source chatbot framework, Rasa. International Journal of Engineering Research **9**(06) (2020)
16. Shawar, B., Atwell, E.: Chatbots: Are they really useful? LDV Forum **22**, 29–49 (2007)
17. Soler Valadés, P.: Agent educatiu multiassignatura: disseny i implementació amb rasa i docker. Tech. rep., Universitat de Barcelona (2023)
18. de Sousa, L.H.H., Trajano, G., Morales, A.S., Sarkadi, S., Panisson, A.R.: Using chatbot technologies to support argumentation. In: 16th International Conference on Agents and Artificial Intelligence (ICAART 2024). SciTePress (2024)
19. Trad, A.: A Conversational Dialogue System for Nutrition And Search Queries. Master's thesis, ULM University (2023)
20. Trajano, G., Engelmann, D.C., Bordini, R.H., Sarkadi, S., Mumford, J., Panisson, A.R.: Translating natural language arguments to computational arguments using llms. In: Computational Models of Argument. pp. 289–300. IOS Press (2024)
21. Zhao, R., Chen, H., Wang, W., Jiao, F., Do, X.L., Qin, C., Ding, B., Guo, X., Li, M., Li, X., et al.: Retrieving multimodal information for augmented generation: A survey. arXiv:2303.10868 (2023)