Sistema de gestión de información sobre el sistema solar

Jan Henrik Sánchez Jerez - 20231020130
Marlon Yecid Riveros - 20231020

Presented to: Carlos Andres Sierra
Universidad Distrital Francisco Jose De Caldas
FACULTY OF ENGINEERING
ING. OF SYSTEMS
BOGOTÁ D.C
2024

The study of the solar system is approached as a complex system, where the need to access and understand a wide range of information has driven the development of specialized tools. In this project, a chatbot has been designed as a solution to satisfy this demand, applying the principles of systems theory. The chatbot, conceived as an information system, has been built by integrating various sources of knowledge, including data collection through extensive research and web scraping. This data, ranging from the origin to the phenomenology of the solar system, has been structured and processed using natural language processing (NLP) techniques. The implementation of the chatbot has been carried out in the Python programming environment, taking advantage of a variety of libraries specialized in NLP and information retrieval. This modular architecture reflects the structure of complex systems, where individual components interact to meet a common goal: to provide accurate answers about the solar system to users. The chatbot, seen as an adaptive system, is capable of receiving user queries and generating responses based on the information previously collected and processed. This capacity for adaptation and feedback reflects the principles of systems theory, where complex systems exhibit emergent and adaptive properties in response to their environment. It is noted that the chatbot is already pre-trained, although not with specific data from the solar system, which provides it with a solid foundation to understand and answer questions

Keywords: Solar system, chatbot, system, adaptability, Natural language processing

1. Introduction

The solar system, with its vast expanse and complexity, remains a subject of fascination and continued study for humanity. In this context, the need arises to develop tools that allow efficient and understandable access to the vast amount of information available on this topic. In line with this objective, this project focuses on the creation of a chatbot specialized in providing answers to various questions related to the solar system.

2. Review of sources:

To carry out a comprehensive analysis of the solar system, it is essential to carry out a thorough review of various sources, since it covers a wide range of aspects. From its origin, marked by the emergence of the Sun, to its evolution and the interaction between the different celestial bodies over millions of years, the solar system has been fundamental to understanding both the formation and development of the planets. This process has been crucial for establishing an apparent balance in the system, allowing, in turn, the emergence of life on Earth, a phenomenon that involves a complex interaction of factors. In this sense, the information provided by NASA has played a fundamental role in our research. As the primary space agency charged with collecting data and carrying out space missions, NASA has been an invaluable source of knowledge for understanding the phenomena that shape our solar system.

3. Methodology

The main purpose of the project was to develop an information management system focused on the solar system. To achieve this goal, we received guidance from Professor Carlos Andrés Sierra, who provided us with the code necessary to implement the chatbot. Below we will detail each step of the code and how it contributes to the functionality of the chatbot

3.1. Libraries used

The libraries that were used for the chatbot are the following:

- `sys`: is a standard Python module that provides access to some variables and functions that interact strongly with the Python interpreter. Use in code : Used for sys.exit() which allows the program to be exited when the user enters .^{ex}it”.

- `langchain`: is a library designed to facilitate the creation of applications that use natural language models. The RetrievalQA module provides a class for building question and answer (QA) systems that combine information retrieval with answer generation. Use in code : Used to create a question and answer object that combines a language model with a vector index to search and generate answers.

- `langchain_community`: extension that includes several document uploaders. PyPDFDirectoryLoader is used to load and read PDF files from a directory. Usage in code : Used to load the contents of all PDF files in a specified directory and return it as a text string.

- `langchain_huggingface`: FAISS (Facebook AI Similarity Search) is a library that enables efficient search of high-dimensional vectors. It is used to create vector indexes that facilitate quick search for information based on embeddings. Use in code : Used to create a vector index from text fragments and their embeddings, allowing efficient searching.

3.2. Code functions

The following functions were defined in the code:

- `load_pdf_data`: This method takes all PDF files in a directory and loads them into a text string. Args: path (str): The path to the directory containing the PDF files. Returns: A text string containing the content of the PDF files.

- `split_chunks`: This function splits a text string into 10,000-character chunks with a 20-character overlap. This is useful for handling large amounts of text more efficiently. Use RecursiveCharacterTextSplitter to perform the split. This method splits a text string into chunks of 10000 characters with an overlap of 20 characters. Args: data (str): The text string to split into chunks. Returns: A list of strings containing the chunks.

- `get_embeddings`: This function obtains semantic embeddings using a specific HuggingFace model (sentence-transformers/all-MiniLM-L6-v2). Embeddings are vector representations of text that facilitate the search and retrieval of information.

  This method gets semantic embeddings for a list of text chunks. Returns: A list of embeddings for the text chunks.

- `get_chunk_embeddings`: This function takes a list of text fragments and their corresponding embeddings, and creates a vector index using FAISS. This index allows quick and efficient searches on text fragments.

- `load_llm`: This function loads a LlamaCpp language model from the Mistral family. The model is downloaded and configured with certain parameters such as temperature, top-p , and others.

  This method loads the LLM model, in this case, one of the FOSS Mistral family.

- `agent_answer`: This function uses the language model (LLM) and the vector index to answer a question. Configures a RetrievalQA object that uses the model and index to search and generate a response.

  This method gets the answer to a question from the LLM model. Args: question (str): The question to ask the LLM model. llm (object): The LLM model.

  Returns: A string with the answer to the question.

3.3. Code execution

The execution of the code is carried out through the `main` function, which is responsible for loading the language model, loading and dividing the PDF data into fragments, obtaining the embeddings for the fragments, creating a vector index and enter a loop where it asks the user for a question and uses the model and index to generate an answer.

3.4. Web scraping

Having already discussed what the chatbot code is, we must focus on how we were going to place the information for the chatbot? The research for the data for the chatbot was carried out in two different ways, investigating some concepts and meanings of some things (such as translation and rotation), in addition, web scraping was carried out on the information to have it faster and more accessible way

To investigate the concepts, everything related to the sun was sought, what it was, what it is made of, the phenomena it causes, its structure and also how it could affect the solar system.

Speaking of the other method of obtaining data, a web scraping process was carried out as mentioned above in order to obtain a large amount of information quickly and efficiently. For this process, two reliable sources were used, which are: - POT - HyperPhysics - https://www.iau.org/public/themes/pluto/ The code that was used to do web scraping is the following: 1. Import of modules : In the first lines, we import the necessary modules to perform web scraping. requests is used to make HTTP requests, BeautifulSoup to parse the HTML of web pages, and Translator to translate text.

import requests from bs4 import BeautifulSoup from googletrans import Translator

2. Translator initialization : We create an instance of the Google Translate translator to be able to translate the text of web pages. translator = Translator() 3. Scraping function : We define a function called scraping that accepts a URL as input. This function does the following:

`Http-get`: Makes an HTTP-GET request to the specified URL to get the page content.

It uses BeautifulSoup to parse the HTML content of the page.

¡p¿tags (paragraphs) in the page content.

Iterates over each paragraph found and prints its text to the console. def scraping(url): page = requests.get(url) soup = BeautifulSoup(page.content, 'html.parser') tagsp = soup.findll('p') for p in ptags: print(p.text) print() 4. Scraping function with translation : We define another function called scrapingt that performs the same operation as scraping , but also translates the text of the paragraphs from Spanish to English using the Google Translate translator.

def scrapingt(url): page = requests.get(url) soup = BeautifulSoup(page.content, 'html.parser') tagsp = soup.findall('p') for p in ptags: text = translator.translate(p.text, src=.^{es}", dest=.^{en}") print(text.text) print() 5. Code execution : After defining the functions, they can be called anywhere in the program by passing the URL of the web page that you want to scrape.

3.5. Discussión

3.5.1. Chaos in the solar system and data uncertainty

The solar system is a complex and dynamic system, where the interaction of multiple celestial bodies generates a considerable degree of chaos and variability. Similarly, collecting data for the chatbot through web scraping may be subject to uncertainty and variability. The data obtained may not always be accurate or complete, reflecting the chaotic nature of the system.

3.5.2. Managing stress in the chatbot and web scraping

It is crucial to consider managing the stress generated by the amount of data being handled and the efficiency with which the process is carried out. The massive amount of information that is collected and processed through web scraping can put significant pressure on computational resources and system responsiveness. Similarly, the chatbot must deal with a variety of user queries and requests, which can increase operational load and affect its performance.

3.5.3. Homeostasis in the chatbot and the solar system r

Homeostasis refers to the dynamic balance within a system. In the case of the chatbot, homeostasis could be related to its ability to maintain stable performance and provide accurate and consistent responses over time. In the solar system, homeostasis is reflected in the various feedback mechanisms that maintain a relative balance between different celestial bodies and phenomena, such as the stability of planetary orbits.

3.5.4. Chaos and homeostasis perspectives

From a chaotic perspective, the solar system can experience unpredictable and non-linear changes due to the interaction of multiple factors, such as gravitational perturbations. However, it also exhibits patterns of long-term stability, such as conservation of mass and energy, that reflect a form of homeostasis on a cosmic scale. Similarly, the chatbot may experience fluctuations in its performance, but its design and adaptability help maintain an overall balance in its functioning.

3.5.5. Response speed

It is important to take this factor into account when evaluating the performance of the chatbot, since the response speed can vary significantly depending on the hardware used. Furthermore, these observations highlight the importance of considering system performance in different environments and usage conditions.

4. 7. Conclusions

From what we have investigated, we can see several things about the behavior of both the chatbot and the solar system:

- **Interconnection**: In a chatbot, the different modules and functions are interconnected so that each part of the system communicates with others to process and respond to user queries. For example, a natural language understanding module can send user input to a search engine, which in turn retrieves the relevant information, which is processed by a response generator before sending it back to the user.
- **Dependency**: Each component of the chatbot depends on others to function correctly. If one of the modules fails or cannot provide the necessary information, the overall performance of the chatbot will suffer. Therefore, the accuracy and efficiency of the system depend on the cooperation and effectiveness of all its components.
- **Adaptabilidad y cambio**: Tanto el chatbot como el sistema solar son sistemas dinámicos que pueden adaptarse y cambiar con el tiempo.
- **Interconnectedness**: In the solar system, celestial bodies, such as planets, moons, asteroids, and comets, are gravitationally interconnected. Their orbits and movements are influenced by the gravitational forces they exert on each other. For example, the Earth's orbit around the Sun is determined by the gravitational interaction between both bodies, as well as the gravitational influence of other planets in the solar system..
- **Dependency**: Celestial bodies in the solar system depend on gravitational forces to keep their orbits stable and predictable. Any significant change in the position or mass of a celestial body can affect the orbits of other bodies in the solar system. Therefore, the stability and balance of the solar system depends on the interaction and gravitational dependence between its components.

- **Property emergence** : In both systems, properties or behaviors emerge that are not evident from the analysis of the individual components. For example, in the chatbot, the ability to understand and answer complex questions arises from the combination of natural language processing algorithms and language models. In the solar system, phenomena such as tides emerge from the gravitational interaction between celestial bodies.
- **Adaptability and change**: Both the chatbot and the solar system are dynamic systems that can adapt and change over time. The chatbot can improve its performance by updating algorithms or incorporating new data. For its part, the solar system undergoes changes over geological periods of time, such as the evolution of planetary orbits or the formation and destruction of celestial bodies.
- **Chaos and stability**: Despite the presence of chaos and variability in both systems, they also show patterns of long-term stability. In the chatbot, stress management and resource optimization can help maintain stable performance despite fluctuations. In the solar system, energy and mass conservation mechanisms contribute to the stability of the system on cosmic scales.

5. References

1. NASA: `https://www.nasa.gov/`
2. HyperPhysics: `http://hyperphysics.phy-astr.gsu.edu/hbase/hph.html`
3. IAU: `https://www.iau.org/public/themes/pluto/`
4. National Geographic: `https://www.nationalgeographicla.com/espacio/2022/10/que-es-el-sistema-solar-y-como-esta-compuesto`
5. National Geographic: `https://www.nationalgeographic.com.es/ciencia/mundo-extremos-llamado-mercurio-planeta-mas-cercano-sol_18620`
6. National Geographic: `https://www.nationalgeographic.es/espacio/que-sabes-estrellas`