
The Assistant

Gökçenaz Akyol¹

Abstract

This project focuses on the development of a deep learning-based virtual assistant, inspired by popular platforms like Alexa and Siri. However, one major issue with these virtual assistants is the lack of customization and personalization options available to users. While these platforms are capable of performing a variety of tasks, they often do so in a uniform and impersonal manner, leaving users feeling disconnected from the technology they are interacting with. The aim of this project is to address this problem by developing a virtual assistant that is more interactive and personalized, with a focus on creating a more natural and engaging user experience. In addition to its core functionalities, the virtual assistant incorporates web scraping capabilities, enabling it to gather information from websites efficiently. Moreover, the system can perform various tasks, such as opening Google, utilizing a user-friendly graphical user interface (GUI).

1. Introduction

In today's modern era, virtual assistants have become an integral part of our daily lives. We rely on platforms like Alexa, Siri, and Google Assistant to perform a myriad of tasks and provide us with convenient functionalities. However, despite their widespread use, there is a significant limitation to these virtual assistants - the lack of customization and personalization options available to users. While these platforms are capable of performing various tasks, they often do so in a uniform and impersonal manner, leaving users feeling disconnected from the technology they are interacting with.

To address this issue, a new project has been initiated with the aim of developing a virtual assistant that is more interactive and personalized. The focus is on creating a natural and engaging user experience that goes beyond the limitations of current virtual assistant technologies.

The primary objective of this project is to create an intelligent and versatile virtual assistant that can comprehend and provide relevant responses to diverse natural language

queries. By utilizing deep learning techniques, including natural language processing and machine learning algorithms, this virtual assistant aims to surpass the limitations of its predecessors and offer a more interactive and personalized experience to its users.

To achieve this, the project leverages high-quality training data, including datasets like the Coached Conversational Preference Elicitation (CCPE) dataset and the Taskmaster-1 dataset. These datasets provide a broad range of user queries, preferences, and task-oriented dialogues, enabling the assistant to understand a wide spectrum of user needs and perform various tasks.

The training process for the virtual assistant involves applying transformers and algorithms, which have demonstrated their efficiency and effectiveness. The training period typically lasts around 2 hours, showcasing the ability of the virtual assistant to quickly adapt and learn from the provided data. By continuously learning from user interactions, the virtual assistant aims to improve its performance over time, engaging in meaningful and personalized conversations while providing tailored insights and recommendations.

2. Related Work

Perhaps the most well-known examples of virtual assistants are Amazon's Alexa and Apple's Siri, which have revolutionized the way people interact with technology in their daily lives. Both Alexa and Siri use advanced natural language processing (NLP) algorithms to understand and interpret user queries, allowing them to provide relevant responses and perform a variety of tasks, from playing music to controlling smart home devices. However, these virtual assistants are primarily designed to perform specific functions and are often limited in their ability to personalize responses or engage in more complex conversations with users.

Other related works include Google Assistant, Microsoft's Cortana, and Samsung's Bixby, all of which use various forms of machine learning and AI to improve their performance and provide users with a more intuitive and personalized experience. However, these virtual assistants also have their limitations and often struggle to engage users in a truly interactive and personalized way.

In recent years, the GPT-2 model, developed by OpenAI, has emerged as a powerful tool for natural language processing tasks. GPT-2, short for "Generative Pre-trained Transformer 2," is a deep learning model that employs transformer architecture to generate human-like text. Its ability to understand and generate coherent and contextually relevant responses has made it a popular choice for various applications, including chatbots and virtual assistants.

3. Approach

3.1. Data Sets

Coached Conversational Preference Elicitation (CCPE) is a dataset consisting of 502 English dialogs with 12,000 annotated utterances between a user and an assistant discussing movie preferences in natural language. It was collected using a Wizard-of-Oz methodology between two paid crowdworkers. In this methodology, one worker plays the role of an 'assistant', while the other plays the role of a 'user'. The CCPE dataset serves as a valuable resource for training and evaluating conversational recommendation systems, particularly in the domain of movie preferences. It captures the nuances of user preferences and the natural flow of conversations, enabling the development of more accurate and context-aware recommendation models.

Taskmaster-1 is a dataset that contains 13,215 task-based dialogs in English, including 5,507 spoken and 7,708 written dialogs. The dataset was created using two distinct procedures. Each conversation falls into one of six domains: ordering pizza, creating auto repair appointments, setting up ride service, ordering movie tickets, ordering coffee drinks, and making restaurant reservations. Taskmaster-1 provides a diverse and comprehensive collection of task-oriented dialogs, covering a wide range of domains. It facilitates the development and evaluation of dialogue systems that can effectively understand and fulfill user requests in various contexts. The dataset enables researchers to train models that can handle different types of tasks and engage in natural and fluent conversations with users.

In the scope of the project, both the Coached Conversational Preference Elicitation (CCPE) dataset and the Taskmaster-1 dataset were utilized.

3.2. Model

The DialoGPT model is built upon the GPT (Generative Pre-trained Transformer) architecture, specifically the GPT-3.5 variant. GPT-3.5 is a transformer-based language model that has been trained on a vast amount of text data to develop a deep understanding of language and context.

The architecture of DialoGPT consists of multiple layers of transformers. Transformers are a type of neural network

architecture that excel in capturing long-range dependencies and contextual information in text data. Each transformer layer in DialoGPT includes a self-attention mechanism and feed-forward neural networks.

The self-attention mechanism allows the model to weigh the importance of different words or tokens in a given context. It enables the model to focus on relevant information and understand the relationships between different parts of the text. This mechanism is particularly effective in modeling the coherence and flow of conversational contexts.

DialoGPT utilizes a decoder-only architecture, meaning it does not have an encoder component like in other transformer models such as BERT. In a dialogue context, the model takes a series of previous messages or turns as input and generates a response based on that history.

During training, DialoGPT uses a method called "unsupervised learning" or "self-supervised learning." It predicts the next word or token in a sentence given the preceding context, allowing it to learn the statistical patterns and dependencies in the text data.

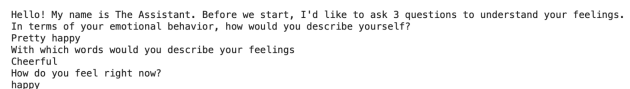


Figure 1. Three questions to determine the emotional state of the user

In the project, the DialoGPT model underwent a fine-tuning process using a specific dataset. The dataset was designed to determine the emotional state of the user by asking them three questions (as shown in Figure 1). Based on the user's responses, the project established a predefined set of emotional categories, which included "happy, sad, angry, romantic, tired, and calm." These categories were used as labels to classify the user's emotional state.

During the fine-tuning process, the identified emotional category was incorporated to optimize the model's performance. By utilizing the emotional category as part of the fine-tuning process, the DialoGPT model was refined to generate responses that were more accurate and contextually appropriate in relation to the user's emotional state.

3.3. Other Key Processes

The project involved several key processes to facilitate the utilization of a trained machine learning model. After the completion of model training, a graphical user interface (GUI) was meticulously devised to enhance user interaction with the model. This GUI was developed using the tkinter library, a widely-used tool for creating user-friendly interfaces in Python.

To further augment the functionality of the GUI, additional libraries such as webbrowser and BeautifulSoup were leveraged. These libraries provided valuable capabilities to incorporate various commands into the interface. For instance, users were empowered to seamlessly navigate to popular online destinations such as Google, YouTube, LinkedIn, and effortlessly request the model's assistance in responding to inquiries conducted through Google searches.

Additionally, the project incorporated the use of the gtts (Google Text-to-Speech) library. This allowed the trained model to effectively communicate with users by converting text into spoken words, thereby enabling a more formal and engaging interaction.

4. Results

The developed project demonstrates the capability to perform three main tasks: executing commands, web scraping, and generating autonomous responses.

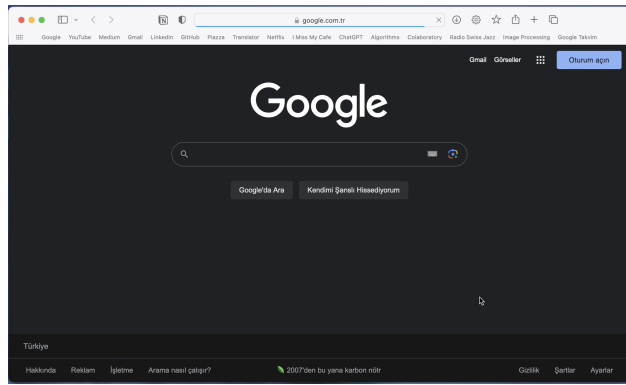


Figure 2. Commands

The first task involves the project responding to specific commands, such as "Open Google," and successfully opens the corresponding application or website, such as Google (as shown in Figure 2).

The second task involves web scraping to retrieve relevant information and answer questions, such as providing the meaning of "World Cup" (as shown in Figure 3). The project showcases proficiency in extracting data from the web and delivering accurate responses.

The third task encompasses the model's ability to generate its own answers (as shown in Figure 4). This means that the project can provide answers or information without relying on predefined commands or web scraping. It showcases the model's ability to generate original and contextually relevant responses based on the input it receives.

It is worth noting that evaluating the performance of such

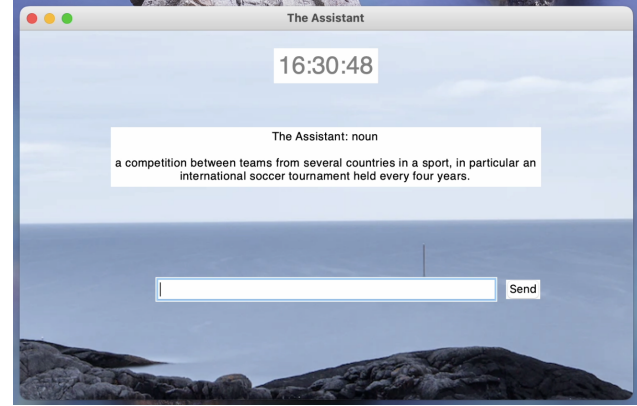


Figure 3. Web Scraping.

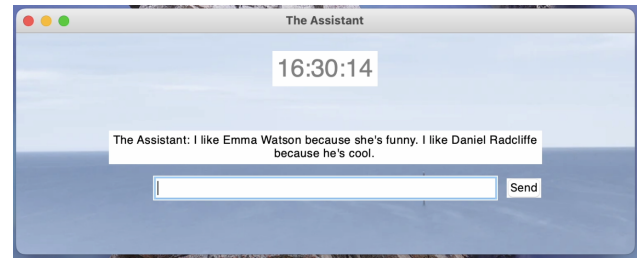


Figure 4. Model generates its own answers

a project can be challenging since there might not be a single metric that encompasses all three tasks. In this case, the evaluation metric for this project is the user experience. Based on feedback from users who have tried this assistant, the overall satisfaction level is reported to be very good. This subjective evaluation allows for an assessment of the assistant's effectiveness in fulfilling its intended purpose of assisting users in executing commands, retrieving information through web scraping, and generating autonomous responses.

5. Main Contributions

- **Graphical User Interface (GUI) Development:** A significant contribution of the project was the meticulous development of a user-friendly graphical user interface (GUI) using the tkinter library. This GUI played a crucial role in enhancing the interaction between users and the trained machine learning model, providing an intuitive and seamless user experience.
- **Integration of Additional Libraries:** The project demonstrated resourcefulness by leveraging additional libraries such as webbrowser and BeautifulSoup to expand the functionality of the GUI. These libraries pro-

vided valuable capabilities that allowed users to effortlessly navigate popular online destinations like Google, YouTube, and LinkedIn, while seamlessly requesting the model's assistance in responding to their inquiries.

- **Google Text-to-Speech (gtts) Integration:** To enhance user engagement and communication, the project successfully integrated the gtts library, enabling the trained model to convert text into spoken words. This integration added a new dimension to the interaction, as the model could effectively communicate with users through spoken language, creating a more formal, dynamic, and engaging experience.
- **Fine-tuning with Emotional Classification:** A notable advancement in the project was the fine-tuning process of the DialoGPT model using a specialized dataset. The dataset included three questions designed to determine the emotional state of the user, resulting in the identification of predefined emotional categories such as "happy, sad, angry, romantic, tired, and calm."
- **Emotional Context Optimization:** Building upon the emotional classification, the project ingeniously incorporated the emotional categories into the fine-tuning process, optimizing the model's performance. This approach ensured that the DialoGPT model generated responses that were not only accurate but also contextually appropriate, considering the user's emotional state during the interaction.

In summary, the project's main contributions encompassed the development of an intuitive GUI, integration of additional libraries to enhance functionality, incorporation of text-to-speech capabilities, and the optimization of the DialoGPT model's performance through fine-tuning with emotional classification. These contributions collectively aimed to provide users with a seamless and engaging experience, delivering contextually relevant responses based on their emotional state.

6. Discussion

Although the project achieves satisfactory results overall, it should be noted that the data set used during model training required careful categorization and sampling to mitigate memory errors. However, occasional biased responses were observed as a consequence of this preprocessing. Nonetheless, the model predominantly produces successful and relevant answers to various inquiries.

It is important to highlight certain limitations of the project. Due to the lack of asynchronous implementation, the speaking and clock functions operate sequentially. Consequently, a new question cannot be posed until the speaking command

has completed, and the clock stops during the speaking process. While this limitation restricts the system's multitasking capabilities, it does not significantly impede the overall performance of the application.

In conclusion, the developed project showcases promising results in executing commands, web scraping for information retrieval, and generating autonomous responses. The occasional biased response due to the preprocessing of the training data set and the limitations surrounding asynchronous implementation of the speaking and clock functions are acknowledged. However, the project primarily delivers successful answers and demonstrates proficiency in handling a broad range of queries.

7. References

- [1] <https://research.google/resources/datasets/coached-conversational-preference-elicitation/>
- [2] <https://research.google/tools/datasets/taskmaster-1/>
- [3] <https://arxiv.org/pdf/2006.15720.pdf>
- [4] <https://arxiv.org/pdf/1911.00536.pdf>
- [5] <https://pypi.org/project/beautifulsoup4/>
- [6] <https://docs.python.org/3/library/webbrowser.html>
- [7] <https://docs.python.org/3/library/tkinter.html>
- [8] <https://huggingface.co/docs/transformers/index>