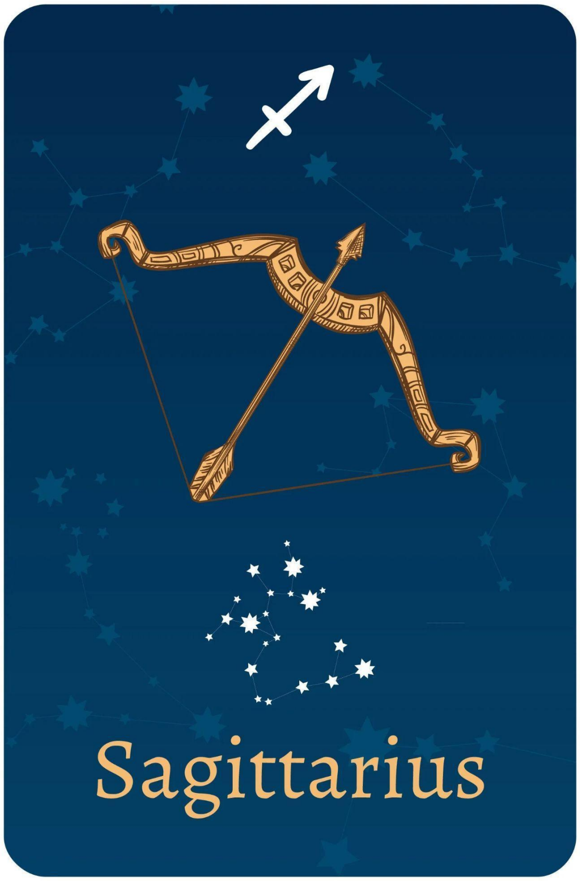
SJSU SAN JOSÉ STATE UNIVERSITY

Department of Electrical Engineering
EE104 - Fall 2023



TEAM MEMBERS

Shivansh Shukla

Contents

PURPOSES AND GOALS

This assignment focuses on building a voice-activated chatbot, integrating speech recognition, natural language processing, and text-to-speech. Participants will use speech_recognition and gtts for essential functions, and incorporate a pre-trained language model like DialoGPT from transformers. The goal is to create a responsive chatbot that understands spoken language, generates contextually relevant responses, and includes features like wake-up phrases and dynamic interactions. The assignment aims to equip participants with practical skills for developing engaging voice-driven AI applications.

HOW TO INSTALL THE PROGRAMS

DEPENDENCIES

To run the program, install the following dependencies using pip: speech_recognition, gtts, transformers, and numpy. Ensure Python 3 is installed. For text-to-speech, adjust the system command (afplay for macOS, start for Windows). The code is OS-compatible, but minor adjustments may be needed. It utilizes the microsoft/DialoGPT-medium model, requiring an internet connection for the first run. Set the environment variable: os.environ["TOKENIZERS_PARALLELISM"] = "true". Refer to the official documentation for troubleshooting.

HOW TO RUN THE PROGRAMS

To run the AI chatbot program, start by ensuring that you have Python installed on your system. Then, install the necessary libraries by executing the commands pip install SpeechRecognition, pip install gtts, pip install transformers, and pip install numpy. If you're using Windows, consider adjusting the system command for text-to-speech playback to start. Once the dependencies are in place, run the program by executing the Python script with the command python your_script_name.py, replacing your_script_name.py with the actual script name. The chatbot will be activated and ready to respond. Use the specified wake-up phrase, in this case, "Jarvis," to initiate interactions. Engage in conversations, ask questions, and explore the chatbot's capabilities. To gracefully exit the conversation and close the program, use commands like "exit" or "close." Ensure an internet connection during the first run for the program to download the pre-trained language model (microsoft/DialoGPT-medium). Following these steps will allow you to seamlessly run and interact with the AI chatbot program. Adjustments can be made based on specific preferences or requirements.

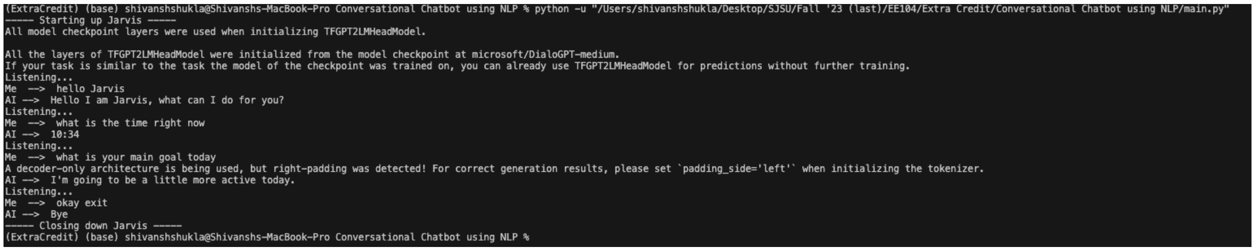
DESIGN ARCHITECTURE

The architecture of the AI chatbot program involves a cohesive interaction between hardware and software components. On the hardware side, the microphone captures user audio, while the speaker plays back the chatbot's synthesized responses. The local processing unit, which could be a computer or mobile device, serves as the core hardware block responsible for executing the Python script and managing local processing tasks. The software logical blocks include the speech recognition module, utilizing the speech_recognition library to convert audio input into text. The text-to-speech module, powered by the gtts library, synthesizes the chatbot's textual responses into audible speech through the speaker. The integration of a pre-trained conversational language model, such as DialoGPT from the transformers library, enriches the chatbot's ability to generate contextually relevant responses. The main program logic coordinates the overall interaction, handling user input, wake-up phrases, and conversation flow. Optionally, cloud-based components, like the conversational language model, may be involved, requiring internet connectivity. This architecture emphasizes the collaboration between different hardware and software blocks to create a responsive and engaging voice-activated chatbot system.

PROCESS & WORKFLOW

The AI chatbot's workflow begins with the user initiating interaction through a wake-up phrase, triggering speech-to-text conversion via the microphone. The main program logic processes user input, utilizing a conversational language model for context-aware responses. Text-to-speech synthesis transforms the chatbot's responses into audible speech. The conversation iterates dynamically, allowing users to inquire, command, and engage. When users decide to conclude the interaction, the chatbot delivers a closing message, concluding the program. Optionally, cloud resources may enhance the chatbot's capabilities. This concise workflow mirrors natural conversation, fostering a seamless and engaging user experience.

TEST DATA



```
(ExtraCredit) (base) shivanshshukla@Shivanshs-MacBook-Pro Conversational Chatbot using NLP % python -u "/Users/shivanshshukla/Desktop/SJSU/Fall '23 (last)/EE104/Extra Credit/Conversational Chatbot using NLP/main.py"
----- Starting up Jarvis -----
All model checkpoint layers were used when initializing TFGPT2LMHeadModel.

All the layers of TFGPT2LMHeadModel were initialized from the model checkpoint at microsoft/DialoGPT-medium.
If your task is similar to the task the model of the checkpoint was trained on, you can already use TFGPT2LMHeadModel for predictions without further training.
Listening...
Me  -->  hello Jarvis
AI -->  Hello I am Jarvis, what can I do for you?
Listening...
Me  -->  what is the time right now
AI -->  10:34
Listening...
Me  -->  what is your main goal today
A decoder-only architecture is being used, but right-padding was detected! For correct generation results, please set `padding_side='left'` when initializing the tokenizer.
AI -->  I'm going to be a little more active today.
Listening...
Me  -->  okay exit
AI -->  Bye
----- Closing down Jarvis -----
(ExtraCredit) (base) shivanshshukla@Shivanshs-MacBook-Pro Conversational Chatbot using NLP %
```

VIDEO RECORDINGS

Recording Title	URL	Notes
EE104 - Extra Credit: Conversational Chatbot	▶ EE104 - Extra Credit: Con…	Demo of Conversational Chatbot Extra Credit

CONCLUSIONS

In conclusion, the primary purpose of this assignment was to explore the integration of speech recognition, natural language processing, and text-to-speech technologies to develop a voice-activated AI chatbot. This goal was successfully achieved by implementing a functional chatbot that recognizes spoken language, generates contextually relevant responses using a conversational language model, and employs text-to-speech synthesis for natural interactions.

Throughout the assignment, I learned valuable lessons in coordinating hardware and software components, managing user interactions, and leveraging pre-trained language models. The success of the project highlights the potential for creating engaging and responsive voice-driven AI applications.

To improve in future iterations, I could consider refining the user experience by implementing more sophisticated natural language understanding and response generation techniques. Additionally, exploring cloud-based solutions for enhanced scalability and accessibility could be beneficial. Access to better resources, such as more powerful language models or increased computational capacity, could further elevate the chatbot's capabilities.

In essence, this assignment provided a foundation for understanding the intricacies of building voice-activated chatbots and offered insights into potential areas for refinement and expansion. The experience gained will undoubtedly contribute to future endeavors in creating advanced and user-friendly AI applications.

REFERENCES

- What Is Agile Reporting? (Definition and How To Complete) https://www.indeed.com/career-advice/career-development/agile-reporting
- Test Report https://strongqa.com/qa-portal/testing-docs-templates/test-report