ASSISTANT POLY

1. THE IDEA

Introduction

For the last project of our fifth and final year at Polytech, we wanted to end our scholarship on a high note by taking full advantage of the latest innovations in machine learning.

Right when we began searching for an idea, something came out that shook the world. In less than a week ChatGPT had already reached over a million users, and made everyone realize that we are entering into a new era.

The idea

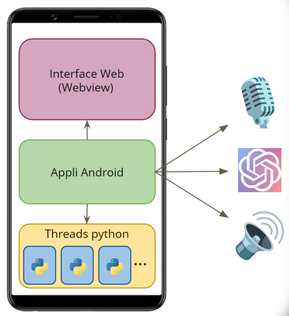
ChatGPT can answer questions, give advice, write emails and even code. A personal assistant that works quickly and tirelessly. In fact it is much smarter than the so-called smart assistants we all have in our phones, but unlike those unfortunately, it is trapped within a web page, unable to interact with our devices.

ChatGPT can't use those predefined functions, but it can do something much more powerful, it can write code. If we give it a little help by executing that code on the device, we can use any function. We just need teach it how to behave like an assistant, and respond to action requests.

There is no doubt that soon every assistant will be upgraded to match ChatGPT's performance, but in the meantime, we have an opportunity to be at the cutting edge of mobile assistants, we just need to take the right technologies and glue them together.

The plan

To put this idea into practice, we want to create a proof of concept, an android app, that executes python scripts in the background based on ChatGPT directives. We need 3 APIs, for speech recognition, speech synthesis, and of course ChatGPT. The interface can be created easily using web technologies. We can even add bonus user experience features.



2. THE PROCESS

Elite INFO team

Keming Zhang and Mirette Guirguis: Front end and android commands

Georges Harrisson Simo: Speech recognition and synthesis, devops

Julian Royet: Back end, architecture

Didier Donsez: the SUPERVISOR

Rude awakening

We engineered a prompt that would make ChatGPT generate python code for android when asked to perform an action, and it worked quite well.

But the entire project relied on the assumption that we could run python code on android. Half way into the project, we realized it wouldn't be possible. We did manage to run python on android, but it could not actually interact with android, making it pointless. It was nothing more than a fancy calculator.

It turns out that ChatGPT had invented this "android python" code. Moreover, the ChatGPT interface was written in python. If we don't run python on android, then we can't use the interface.

We had to delete all that work and find alternatives

API chaos

To spice things up, there is no official ChatGPT API, the interface we use is made by a single developper in his free time, and he has to constantly adapt to OpenAI's internal changes. Of course, every time the interface is updated, it changes completely, and the code that has been written must be rewritten.

Standard Android pain

Android development isn't known as particularly fun, as always a lot of time is spent trying to navigate the vast and deep waters of the Android standard library, with all its obsolete references and compatibility issues. It's not easy to get into.

3. THE SOLUTIONS

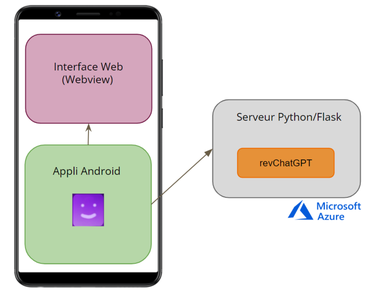
Backup plan

If we can't run arbitrary code, we can still use a more traditional approach: predefined functions. Instead of triggering them based on a specific keyword by the user, we can give ChatGPT a list of command to use at will. That way, we live the interpretation of the request to ChatGPT.

We created commands such as VIDEO, to play a video on youtube, CALL to call someone, SMS to send a message, etc. ChatGPT can pass parameters with these commands, to specify a search query or phone number

New server

To solve the python on android issue, we had to create a separate server. Made with Python and Flask, this server allows us to use the ChatGPT API, and will be used as an intermediary between the Android app and ChatGPT. It is currently hosted on azure cloud.



Local services

Instead of using speech recognition and synthesis APIs, we use local services already installed on the device. For example some Android phones will use Google's algorithms and APIs internally for speech recognition. Using those services makes it much easier, and free, to work with speech, unlike normal APIs.

Advanced research

Because our initial goal failed, we set our eyes on a new one, inspired by Bing Chat. Can we get Poly to gather information automatically when needed ? To accompish this task we use the command system, and another python script that grabs text data from web pages. When Poly uses the SEARCH command, we search the web, grab the text from the results, send it to Poly and ask for a summary.