Voice User Interface: Questionnaire App

Aliaksandr Hladkou
Dr. Alvin Grissom II
CS-474A: Human Computer Interaction
Ursinus Colloge

February 28, 2020

Abstract

In the modern world, there are many different user interfaces presented to users. While there are many solutions already presented, there are still issues may be found even if using most common/famous interfaces. The purpose of this project is to demonstrate how automatic speech recognition (ASR) and text-to-speech (TTS) can be used to provide enjoyable usage and discussing possible issues.

1 Introduction

Presented program (Questionnaire.py) is a program that is completely and only navigable by user's voice. The purpose of the program is to let a user complete a set of several questions to identify how well that user can answer questions. Once the program started, a user will both hear and see program's output to navigate a user for each step. In addition, a user will also see a navigation tab with commands to proceed. In the begin of the program, a user will be asked to login or create new user in order to provide personalization. Once a user is identifies, he will be prompted to complete questionnaire. After completing all questions, a user will be giving a score and automatically signed out.

2 ASR, TTS and Decisions

While integrating ASR into current program, there were many important decisions made in order to provide easy and not confusing interaction.

2.1 ASR

In order to let user understand when to start speaking, it was necessary to provide some kind of feedback. In this case, the decision was made to provide carefully chosen sound every time when the program starts listening. If ASR does not recognize user's speech it will randomly take phrases to ask a user to

repeat. Number of tries to repeat is limited. Additionally, ASR provides visual feedback on whatever user says.

2.2 TTS

In order to make closer interaction between a user and the program, the decision was made towards providing navigation for each step a user have to take (say). It was also important to let the program say information only in appropriate places, so the user does not receive too many information. Therefore, the program speaks only the program moves towards the next step or prior user's action.

2.3 Visual

In addition to all information provided to user, it was also important to choose what information to display. As described above, a user does not hear all options. Therefore, the information about navigating the program is wisely distributed into visual and TTS. Furthermore, all information said by a program will be displayed, so a user will not get lost if sound will not working or if a user will be distracted. Finally, a user will also see whatever he says in same purposes.

2.4 Other Decisions

In every stage of the program, if a user does not provide any speech, then the program will automatically exit after specified amount of tries. Every time an issue occurs, a user will hear an appropriate feedback.
During the test, a carefully chosen sound was also added to provide a feeling of moving to the next step. Furthermore, the program contains the personalization, i.e. every user has a unique nickname. To approve a user, a program also asks for keyword. The following flowchart demonstrates the behavior of program:

3 Issues and Further Development

The are many decision were made to provide save of errors and enjoyable to use program. However, there are still several issues exists in order to make this program to be completely independent from text or mouse input/output to function. For example, chosen for this project speech recognition API does not always recognize properly user's speech. In addition, it is difficult for this ASR to recognize single words or short phrases. User's accent may also be the issue to recognize. These issues often lead to increasing time to complete each task. For example, clicking to select an answer or option will provide a significant difference of time. Furthermore, the security when providing the personalization is also an issue. The current program uses user's keyword as a password. However, a user have to pronounce it which excludes the security.
In order to improve current program, it is necessary to include a function that will take a specific letters of word or parts of phrases from user's speech, calculate

the percentage of matching correct result, and then offer answer matching highest result.

4 Conclusion

This projects demonstrates that while it is possible to integrate ASR into interfaces, there are still many issues exist. The demonstrated issues can also be addressed to the modern ASRs in production. Even if they are good for completing certain functions, at the present moment, it is not possible to make a "hands-free" programs.