CSCE 5290: Natural Language Processing
Project Proposal

Google Collab Speech to Text

Group Members

Solomon Weatherby
Matthew Blair
Jackson Johanek
Thomas McCullough
Github

Goals and Objectives

Motivation

The speech recognition subfield of computer science has included some of the most innovative technologies that humans have developed over the past half-century. For people who have problems with speech or sight, speech recognition tools provide extraordinary capabilities that would otherwise be very difficult if not impossible. We wanted to start this project so that we can help those who have these impairments with their ability to use Google Colab (a free Juypter notebook environment in which anyone can write and run Python code through their browser without requiring a setup). Furthermore, speech recognition software tools can increase the productivity of the user, perform faster than you can type in some situations, and reduce repetitive tasks (O'Brien). We hope that upon completion of this project, we've not only learned the essence of how speech recognition works within programming, but also that we've provided a useful tool for those who could benefit the most.

Significance

This project is important because of its relevance for those who have a vision impairment. Voice to text is an interesting NLP problem on its own, however this project is an extension above that problem, by integrating voice to text with google collab we can offer another way for programmers to use the IDE. Furthering the programming community as a whole, as well as being a good service for the visually impaired.

It's imperative to accomplish the project because currently we believe there are no tools designed for this purpose to help programmers specifically. On top of the fact that we may be able to potentially improve all Speech to Text functions.

Objectives

Our primary objective is to create an interface through which an end user could control a Google Collab session with their voice. We will accomplish this incrementally, with deliverables possible for every milestone. The milestones are as follows:

1: Create a front-end interface. This will initially be a proof of concept. The interface will be voice controlled, and will respond to speech commands from the end user.

Requirements: Interface provides speech prompts for end users. Interface accepts speech commands from users and acts on them. Optional goal: Use the Amazon Alexa API to implement the interface.

Deliverables: Either a rudimentary Alexa skill, or a python based interactive application.

2: Create a simple text processor to manipulate the Google Colab / Jupyter Notebook directly.

Requirements: The text processor should have several callable functions to perform basic actions on a Jupyter Notebook. Initially actions such as renaming and adding cells to the end of the document, then expand to include adding, editing and deleting arbitrary cells from the notebook. An optional goal is support for naming cells and regions of the notebook for easier manipulation of the document.

Deliverables: A python module containing a series of callable functions.

3: Create a Google Colab reader. This will be an intermediate step necessary for full integration with the front-end application.

Requirements: The reader should take a valid Google Colab notebook, parse the file for blocks, and turn them into plaintext that can be sent to the front-end application. The reader should not only be able to convert the document into plaintext, but also allow for access to arbitrary sections of the document. An example would be accessing an arbitrarily numbered cell and reading its contents.

Deliverables: Python Module with functions for converting a Google Colab / Jupyter Notebook, and functions to access discrete regions of the notebook.

4: Integrate reader into front-end application

Requirements: Add functionality to read the entire notebook, or a given cell by a given number.

Deliverables: Alexa skill or Python application capable of basic read queries for the document. Application will respond to verbally given commands.

5: Integrate basic editing commands into the front-end application

Requirements: Application will respond to verbally given commands to rename notebook, add text cells and code cells, and move cells up and down

Deliverables: Update to front end application capable of basic notebook editing functions

6: Integrate advanced editing commands into front-end application

Requirements: Application will respond to commands to delete cells, edit text cells, and run code cells

Deliverables: Update to front-end application with advanced editing features

7: Create a Python interactive shell capable of constructing code blocks from a description of the code.

Requirements: Shell should be capable of naming arbitrary variables, performing mathematical operations, and printing to console. Optionally, the shell should be capable of declaring loops, importing libraries, and both declaring and calling functions.

Deliverables: Python module that provides interactive shell when called.

8: Integrate all functions accomplished in task 7 into the front-end application. This is a separate task, as the number of possible input commands will grow significantly.

Requirements: The front end application should be capable of handling a stream of speech input, parsing it for the commands defined in milestone 7, and constructing a code block to meet the parameters.

Deliverables: Finished application capable of accepting speech commands and interpreting them to edit a Google Colab / Jupyter Notebook.

Features

Our program's first feature will be the integration of speech to text that allows the program to interact with Google Colab through speech as the user provides instructions. The next feature will highlight cell manipulation through speech to text. Meaning the user can add, edit, delete, and move cells through speech; along the way other features will be file and text manipulation which will help with formatting issues that the user will like specified. Finally the last feature and the most notable and useful will be speech to text coding within the cells. This will allow the user to have a conversation with the program as they build it together as questions are thrown back and forth to properly build the Python program that is needed.

References

O'Brien, S. (2021, May 20). *Voice recognition*. RingCentral UK Blog. Retrieved October 7, 2022, from https://www.ringcentral.com/gb/en/blog/definitions/voice-recognition/

https://github.com/Crimso777/Mercury-Notebooks