Elliott Rose

Prof. Huang

COSC 625

02-11-20

Automated Research Assistant – Initial Plan

Researching the many needed items when implementing a project often consists of many small steps, with each step only requiring a small bit of time. But often, larger projects require many instances of these research tasks, and the sub-tasks (opening a browser or file, entering a website, reading through information, etc.) end up consuming much of our time. To combat this, I would like to create a voice-operated research assistant program that will take orders and read text back to the user. The assistant should then take further commands to search within and navigate through the text in real time.

To accomplish this build, I will primarily use packages available for the Python language. A Natural Language Processing framework will be needed to parse and identify any queries or commands posed to the program, with regular expressions being used to help identify sentences and paragraphs. A package such as Spacy or possibly even one of the many new forms of BERT, created by Google, might work to suit the NLP needs. To handle the language portion of the code, I will use an open source package available for python. Once the main backend functionality of the assistant taking a command and producing an answer is achieved, this voice functionality will need to be laid on top, finalizing the application.

At this point in time, I have already started some basic, proof-of-concept work in python, scraping data from the web to answer basic questions posed to the program. This functionality needs to be further built out to include navigation of the retrieved file or webpage data and tested for errors. I believe

that this will be the most difficult portion of the problem, and I hope to be finished with this portion of the project by the mid-term progress report. From there, I will need to couple the researching portion of the application to the voice-to-text engine. For this application, I foresee utilizing keywords for specific processes, such as: “define stalactite,” or “read me my SQL notes.” Further, once the program is in its reading phase, the program should be directable with further commands, such as: “next paragraph” or, “start from the beginning.” These commands should allow the user to quickly access information without touching the keyboard.

I also would like to ensure that the program keeps the user updated, more like a human assistant, especially when searches are inconclusive or taking longer than expected. Comments like, “I’m still looking for this information,” and “the internet connection isn’t stable, I’ll get back to you,” would be more appealing than a program that takes your command and then lets you wait for an unspecified amount of time. This will require working with the internal clock in the computer to find the appropriate updating timings for these types of comments. Additionally, functionality that allows the user to program specific phrases to do specific tasks would be beneficial to further building out the assistant. However, because this project is on a restricted timeline, and fairly large for a single person over a semester, these functionalities will have to be added bonuses if there is available time left in the semester after completing the core functionalities.

To attack this project, I would like to use some of the agile techniques we have already learned about. For example, I am going to focus on Test Driven Development so that I have a clear goal to shoot for in completion of the project. Also, I plan on working in a sprint like fashion, focusing on functionality first, and then building out complexity as the timeline moves along. Further, instead of reinventing the wheel, I plan on using well built packages, frameworks, and software so that I can quickly move through the development process to accomplish the goal at hand.

Bibliography

Sommerville, Ian. Software Engineering. 10th ed., Pearson Education Limited, 2016.

"SpeechRecognition." PyPI, pypi.org/project/SpeechRecognition/.

"SpaCy · Industrial-Strength Natural Language Processing in Python." · Industrial-Strength Natural Language Processing in Python, spacy.io/.

Huggingface. "Huggingface/Transformers." GitHub, 14 Feb. 2020, github.com/huggingface/transformers.