CSCE 240 Chatbot Final Project Report

Brian White

Requirement:

The requirement for the project was to first pick out a South Carolina representative from the South Carolina house of representatives list. I then had to come up with a way for my program to look at the webpage about that representative and extract specific information about that representative. Next was to design a chatbot that could take any input from a user with the goal of answering questions about the representative with the information extracted from the webpage.

Specification:

When doing this assignment, I made sure to at the very least fulfill the requirements given to me. I created a chatbot that is able to take any input from a user and will answer basic questions about the representative I chose. As for how to get this done, I had a lot of freedom to choose how I wanted to do it.

Development highlights:

I made my program in Java, and I found a library called jsoup which is designed to look at an HTML file and has functions to retrieve data by using certain tags in the HTML code. My hope was that these tags would be the same for every representative's page. They were not. Most of them were the same, but a few were off by a numbers or letters. It wasn't the best solution, but it was what I had to turn in and it worked for my case. If I had to redo it, I would definitely choose a different method.

This program uses a "Representative" class, and it's information comes from the 3 data classes: PersonalInformation, ContactInformation, and CommitteeAssignments. The data for the data classes comes from the DataLoader class. The data loader uses static methods to scan a text file which is created from the extractor and send hash maps of data to the data classes. In the UI loop, the program compares each user input against all possible predefined inputs and assigns a confidence level between 0 and 1 using the levenshtein distance algorithm for each possible input. If the highest confidence level is greater than or equal to 0.5, then the program responds to that input accordingly. If the highest confidence level is below 0.5, the corresponding prompt is displayed, and the user is told to rephrase their question.

If the program is run with no arguments, it runs the chatbot loop. When the chatbot loop is started, a text file is created which logs the questions from the user and the response from the system. Every time a question is asked, the number of user utterance is increased by one, and every time the system can confidently respond to a question, the number of system utterance is increased by one.

If the program is run with the argument "-summary", the program calculates statistics based on all logged chat sessions and outputs a summary of all chat sessions.

If the program is run with the argument "-showchat-summary <num>", the program calculates statistics based on a given chat session and outputs them.

If the program is run with the argument "-showchat <num>", the program prints the chat log for that session.

Reuse:

I did not reuse anyone's code for this project. As for making my code reusable, I could have done better. Mainly in the extractor program. I have not tested many different cases, but it seems that my extractor program only works for my specific representative. However, if I were to improve the extractor program, the rest of the program should be able to function with any information.

Future work:

The biggest change that would need to be made is to use regex expressions rather than an external library to gather the information about the representative. It works very well in this scenario, but if I wanted to switch which representative the chatbot answered questions on, I would have to modify the code for it to work. Ideally the program should be generic enough to work for any representative, and I believe that using regex would work and should have been my go-to, but I was lazy and didn't want to learn it. Furthermore, overtime the format of the website is likely to change and the code would need to be updated with it in order to insure all of the necessary information is extracted.