Cody Miller

CSCE 240 Project 6

Introduction

The given problem for this assignment was to create a functioning chatbot that would be able to respond to given input that is on the topic of specific companies 10K filings. After the chatbot determines the subject of the input, it would search for the company 10K text file, and respond with the corresponding section of the 10K. For example, the chatbot would have to respond to "Unity risk factors" by outputting the risk factor section of Unity's 10K file to an output file.

I have worked with some generational algorithm AI and Unity machine learning in the past, but that was only for the purpose of having them learn to beat video games. While this project seems easier, it is more actually complex because it requires providing input to search a document for and then having the AI generate specific text output. The biggest difference from my gaming projects is trying to build an interactive system, instead of turning loose an AI algorithm on a game to learn by itself.

Solution

The objective of this chatbot is to be able to accept any user input and be able to return information from a company's 10K file. The two companies supported in my project are Unity and Tupperware.

The chatbot is to be programmed entirely in Java and should be able to view 10K files from companies Unity and Tupperware. To understand the user's input, I will use a slightly variated version of a formula called "Levenstein distance." After decoding the input, it will determine the correct section of the 10K that the user wants, parse through the 10K file, and print the requested information to an output file.

Components

- Project 1 – extractor
    - Project 1 – extractor is where the base of the project began. In this project, I copied and pasted the 10K files for Unity and Tupperware into a data folder, and parsed these files to get the character count, word count, line count, and part counts. I would then output this information into the output.txt file.
- Project 2 – processor
    - Project 2 – processor is where the original chatbot started to take shape. We used the fundamentals developed in the previous project and added a little to it. In this project, functionality is added where you can type part/item names into the program, and it will output that section of the 10K file into the output.txt file.
- Project 3 – ui
    - Project 3 – ui allows the user to add their input into the console instead of the input file to receive output to the output.txt file.
- Project 4 – userintent2querymapper
    - Project 4 – userintent2querymapper adds some extra functionality for the user. Originally, you had to type very specifically the part/item name for the program to give the desired output. Now, after this project's implementation, you can type

something vaguely like the part/item name and the program will recognize what you are trying to say. I used a slightly modified version of Levenstein distance, which counts the number of unique letters and different letters and gets the ratio between those two values. After determining what the user tried to say, it would output the correct section of the 10K file to the output.txt file.

- Project 5 – sessionlogger
    - Project 5 – sessionlogger allows the user to ask about chat statistics. One of the example commands consists of sessionlogger-summary, which will output statistics of all chats, such as total elapsed time, total user messages, total machine replies, etc.

The best project implementation of the five would likely be Project 5 – sessionlogger. This is the one that I spent the most time fixing, separating files into classes/methods, and organizing my code to be more reusable in the future. My implementation of the chat session logger also hasn't had any errors in testing, which is why I would say it was my best implemented project.

Solution Evaluation

After testing the chatbot with 20 different inputs and examining the output, the following statistics can be said about the performance of the chatbot...

- Precision (Correct answers / Total answers received): 69.23%
- Recall (Correct answers / Answers stored): 81.82%
- F1 Score: 75%

Overall, the chatbot works sufficiently with a 75% F1 score, and gives correct output 100% of the time if the input is formatted correctly.

Code Reuse

- Project 1: Reused code from Jacob Walton for project 1, as I programmed mine in Python before I switched my project language into Java. Mostly easy code reuse, I only needed to change some values to make it be able to be implemented into my final chatbot.
- Project 4: Reused code from Ethan Hammer from project 4 because his chatbot's accuracy was a little better than mine. I copied his regular expressions into my project, as he was more thought out and specific.
- Project 5: My code was reused by Ethan Hammer and Jacob Walton. The code that was taken for both was just a few methods, like writing to the csv file, and my regular expressions for determining the user's input.

Discussion

This project can be used to look through a company's 10K file without having to open the actual file and provides an easy-to-use interface for the user. While this project isn't very significant in its usability, most of the significance comes to the knowledge I gained while working on this project, such as using regular expressions and more advanced file parsing techniques.

Building the solution was a great learning experience for me, and it taught me a lot about programming techniques that can be used with Java or really any other programming language. I also learned about reusing other people's code, which I found to be more challenging than I thought, as others laid out the projects differently and have different solutions to issues than what I did.

In the future, I expect to continue working with AI, whether it be with ChatGPT or some Unity machine learning project that I decided to work on, and this project has definitely advanced my knowledge in this area of work.

Conclusion

The development of the chatbot has been a significant learning journey. Through the various project phases, from initial data extraction to enhancing user interaction and refining accuracy, the process has deepened my understanding of programming techniques, particularly in Java, and highlighted the importance of code reuse and collaboration.