Evaluation and Report

System Description:

My chat bot is a comprehensive system that talks to users about all things star wars related. The chatbot incorporates numerous NLP techniques such as **NER, POS Tagging, Tf measures, sentence parsing and pattern matching, sentiment classification etc**. Below, I have detailed the step by step creation of my chatbot.

1. Firstly, I created the knowledge base for my chatbot. I did this by crawling on 50 web pages for star wars information and saved them all into text files. From these text files, I extracted the top 40 key items about Star wars. This was **done using the tf (term frequency) measure**. Once I obtained the key words, I collected the most important sentences for the top 13 key words and stored the entirety of the information in a dictionary and dumped it into a pickle file: 'kb.p'.
2. This next step involved creating the basic outline for my chatbot. I went ahead and created a basic program with a while(True) loop to keep running until the user specified to quit. I then build some basic functionality to the chat bot, such as greeting the user and obtaining their name (basic form of input), etc.
3. Right after creating a skeleton of the chatbot, I created a new User class. Through this class, I defined my user model and created all the attributes and functions required for a user. The attributes are: **name, likes, dislikes, and the personal conversation log**. For each modifiable attribute (like and dislike) I created a get, modify and remove method.
4. Once the user model was ready, I created an empty pickle file, 'user.p' which would store the list of users and their personal information. In my chatbot, I built all functionalities to retrieve all the information of a user based on their unique identifier: the name.
5. The next logical step was to identify the name of the user through casual dialog. I achieved this by utilizing **Named Entity Recognition** using an inbuilt library called spacy. I extracted the label of only the words which had the label of 'PERSON' and then extracted the name of the user from that text.
6. After identifying the user, I had access to all of their information. I then made a connection to OpenAI's API to access ChatGPT's expertise on star wars information, in case my knowledge was not sufficient to answer the given question.
7. I started coding all the important details in the while(True) loop. I started with taking input from the user. If the user inputted 'QUIT' then I would just end the program. If not, then I would go ahead and parse through this input. Firstly, I would **analyze the sentiment of the user input utilizing nltk's inbuilt sentiment analyzer (VADER)**. The next step would be to check if the sentence

was negative or positive. If the sentence had a positive connotation(I specified a threshold of score['pos']>0.5 for this), then I add the topics in the sentence to the likes of the user. If the sentence had a negative connotation(I specified a threshold of score['neg'] >0.5 for this), then I add the topics in the sentence to the dislikes of the user. If a user's previous likes became a dislike, I would take care of this in my code as well by removing from dis like and adding to likes and vice versa.

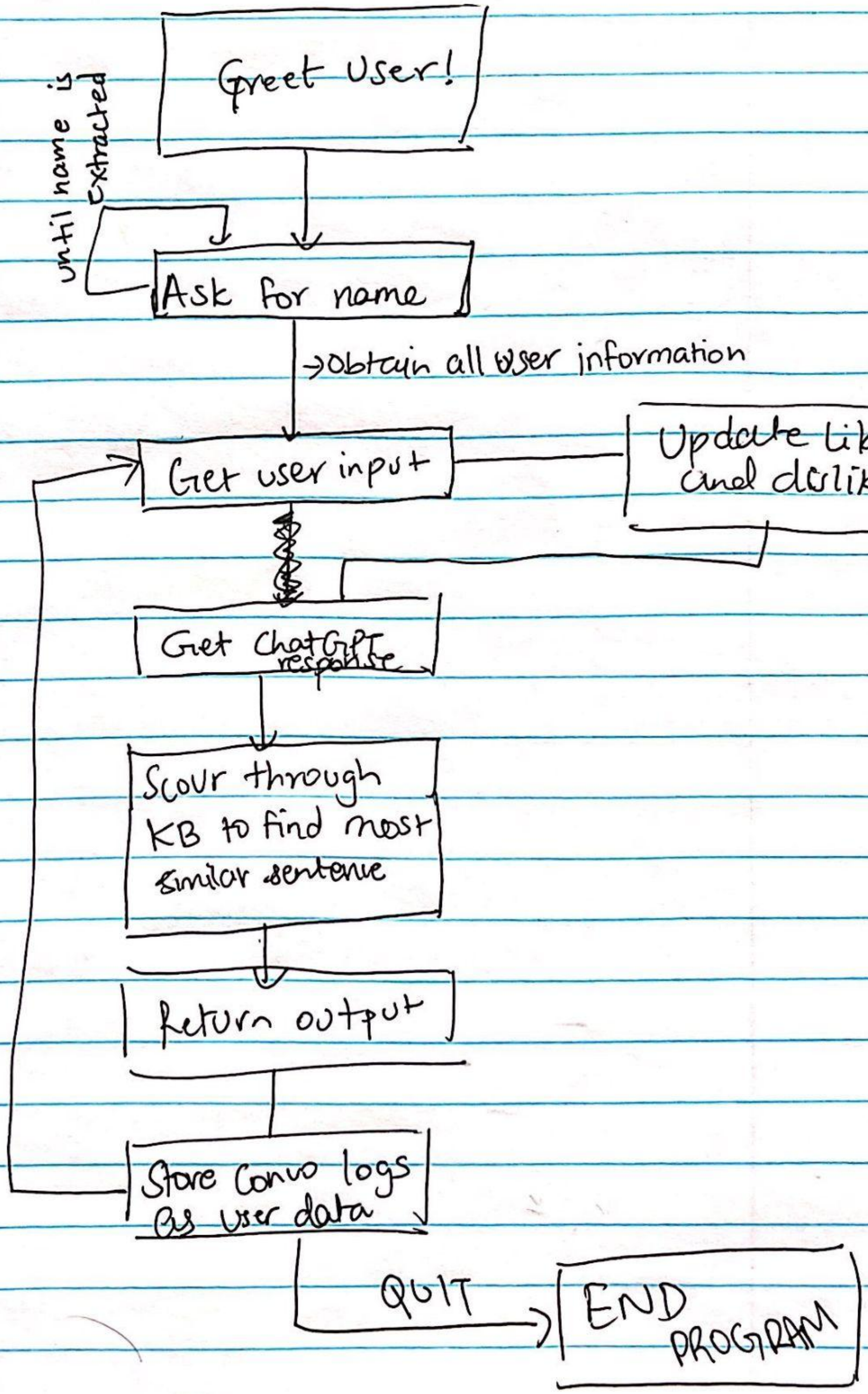
8. To figure out which words in the user's sentence are the main subjects, I utilized **POS tagging.** I would tag each word in the sentence, and then only retrieve the words with the tags NNP or NN. This filtered out all other types of verbs such as helper nouns, verbs, adverbs etc. **I then filtered out the** words which were classified as NN but not really nouns, by **parsing through each of these words individually**. I then made a quick addition to my code to **only account for words which start with capital letters**, as this would mean that they are proper nouns.
9. The next step was to send the user input to ChatGPT and get a response from ChatGPT. Once I obtained a response from ChatGPT, I then **performed POS tagging on the response** once again. I extracted words that might be related to star wars from this, by utilizing methods similar to the ones discussed in Step 8. I matched these words with my knowledge base and looked for sentences in my knowledge base that might have the most similar meaning contextually to ChatGPT's response.
10. The way I did this was by checking the similarity measure between each potential sentence in my knowledge base to ChaGPT's response. **I utilized the cosine similarity measure to check the similarity between two sentences by comparing the word vectors/word embeddings** and exploring the multidimensionality of a word's meaning representation.
11. Then I would compare how high the similarity is between ChatGPT's response and my most similar sentence from the knowledge base. If the sentence from the knowledge base did not cross the threshold of similarity that I set, then I would just go ahead and return the response from CHATGPT, else return the response from my knowledge base.
12. This concludes the coding for the chatbot experience. The next step would be incorporating the user information with their likes and dislikes. After the user quits the chatbot, the program automatically saves the latest version of the users into the pickle file and also prints it to the log so the user can take a look at their new user information.

ChatBot Logic Tree: On the next page!

Ashwin Somasundaram

CS4395

Chatbot Logic Tree


Greet User!
until name is extracted
Ask for name
→obtain all user information
Get user input
Update likes and dislikes
Get ChatGPT response
Scour through KB to find most similar sentence
Return output
Store Convo logs as user data
QUIT
END PROGRAM

Sample Dialog Interaction with Starbot:

```
Starbot: Hello! Welcome!
Starbot: Before we begin, could you please tell me your name?
You: Ashwin
Starbot: Welcome back Ashwin!
You: Who are you?
Starbot: I am Starbot, an assistant who can help answer
questions only about Star Wars and talk about your likes and
dislikes only about Star Wars.
You: What is your name?
Starbot: My name is Starbot, I am an assistant who can help
answer questions only about Star Wars and talk about the
user's likes and dislikes only about Star Wars.
You: Nice to meet you!
Starbot: Likewise! How can I assist you with Star Wars?
You: who is darth vader?
Starbot: Darth Vader is a fictional character in the Star
Wars franchise. He is a Sith Lord who was once a Jedi named
Anakin Skywalker, but was seduced by the dark side of the
Force and became a member of the Sith.
You: what star wars movie would you reccomend?
Starbot: I would recommend Star Wars: Episode V - The Empire
Strikes Back.
You: nice, i will check it out
Starbot: Great! You won't be disappointed.
You: QUIT
Name of most recent user: Ashwin
likes of most recent user: {'Nice', 'JarJar', 'Anakin'}
dislikes of most recent user: {'Padme'}
```

```
This user's convo log has also been saved. In order to view
it, uncomment the line that prints out the convo log
```

Appendix for Knowledge Base:

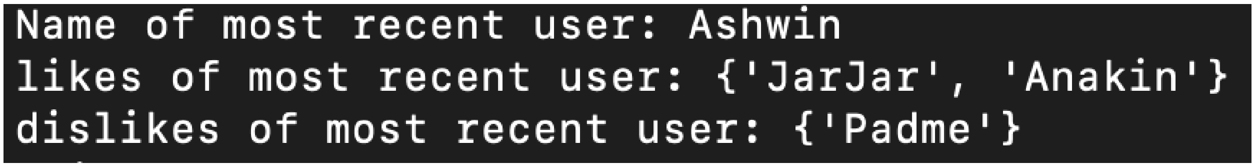
There are numerous key words in my knowledge base. I have listed each one of them below.

- Empire
- Republic
- Mandalorian
- Skywalker
- Kenobi
- Anakin
- Lucas
- Trilogy
- Vader
- Yoda
- Lightsaber
- Luke
- Solo

Each key word has about ~15 sentences associated with it, forming about a knowledge base of size 225 sentences.

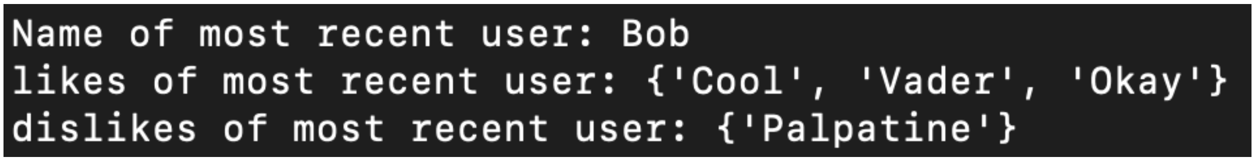
Appendix for Sample User Models:

The first user is me:



```
Name of most recent user: Ashwin
likes of most recent user: {'JarJar', 'Anakin'}
dislikes of most recent user: {'Padme'}
```

Here, we can see that the user likes JarJar and Anakin. The user dislikes Padme.
The second user is Bob:



```
Name of most recent user: Bob
likes of most recent user: {'Cool', 'Vader', 'Okay'}
dislikes of most recent user: {'Palpatine'}
```

We can see that in the case of the second user Bob, sometimes some capitalized words are also attributed to be likes.

Chatbot Evaluation:

Strengths:

- The chatbot responds properly to each of the user's queries.
- The connection is smooth, and the answers to each question are definitely relevant.
- The chatbot can remember the likes and dislikes of the user. It can also store multiple user models. The chatbot can also personalize information based on the preferences of the user.
- The chatbot stores individual user information as a part of the conversation log.
- The chatbot incorporates numerous NLP techniques such as NER, POS Tagging, Tf measures, sentence parsing and pattern matching, sentiment classification etc.

Weaknesses:

- The chatbot can sometimes add non-nouns as your likes or dislikes.
- The chatbot can time-out of the OpenAI api, if there are too many requests within a minute.
- The knowledge base does seem to be very sparse in terms of quality information, thus affecting the quality of information that might be sent to check similarity with OpenAI API's responses. A better knowledge base could easily mitigate this concern.
- Sometimes, the NER technique is unsuccessful at recognizing names, when the names might be classified as cities for example.

Conclusion: I had a lot of fun building this chatbot and believe that it was a great learning experience. In terms of future goals, I would continue to refine the knowledge base and add in a machine learning model and train it to better predict the similarity between sentences and generate responses on its own.