Kyle Chan kxc180021
Ryan Banafshay rcb170002

Chatbot Report

For this project, we decided to design and implement an interactive chatbot from scratch. We decided against using any supplementary platforms, such as Google's DialogFlow or Amazon Lex, as we felt those frameworks did a majority of the work leaving us with very little to code on our own. The chatbot is specifically built to communicate with users about topics related to astronomy. The user can ask questions or make direct statements to the prompt and the chatbot will formulate a response based on our knowledge base. The system will keep track of different users by making use of a user model. The user model stores each user's name while also remembering the different topics the user is interested in. This information is then stored into a pickle file. We used a variety of natural language processing techniques in the implementation of this project; such as cutting out stopwords, lemmatizing sentences for precision, analyzing sentiment scores from user responses, etc.

The system initializes by first opening our intents JSON file. This is how most of the responses outside of the knowledge between the bot to the user is determined. Within the file are a collection of 'intents' with different categories such as greetings, goodbye, welcomeback, thanks, and dislikes. The bot will determine which intent is relevant based on a list of user responses. If there is a certain threshold of similarity between the user response and the list of possible patterns within the intent is reached, then the chatbot will then use that intent's list of responses to communicate back to the user. This calculation is done in the classify function. This function uses a model with the x and y values representing the words and the different categories/intents respectively. Those values are then inserted into sklearn and the model will then assign which intents are most relevant in ascending order. The intents JSON also contains a list of context_req and context_set. The category context_req describes what preconditions must be met to access this intent. The prerequisites are represented by previous intents

accessed by the user. The context_set list will set the context once the intent is reached and used.

Once the JSON is loaded, the model is then loaded in. This model is created using sklearn. For our training data, we used a variety of natural language processing techniques to give us better results for our model. We eliminated all stop words except for no, not, can, and are. Every word was then subsequently stemmed and lowercased. Finally we removed all duplicates leaving us with an improved set of data to work with. The user data, which is represented by a continually updating pickle file, is then loaded and initialized allowing for the chatbot to begin the conversation by asking for the user's name. When the user starts communicating with the chatbot, it determines what intent is most relevant and bases a response on that intent. For simple questions that are outside the scope of our knowledge base, the communication between the user and the bot is done exclusively with the intent system. Once the system recognizes the user is starting to ask questions on the topics of astronomy and blackholes, we flag that we must go into our knowledge base to give a response. When this occurs, we save and then lemmatize the particular subject the user is asking for. We then take that lemmatized word and search for that topic in the knowledge base. From there we extract the sentence that is relevant to the question and give it back to the user.

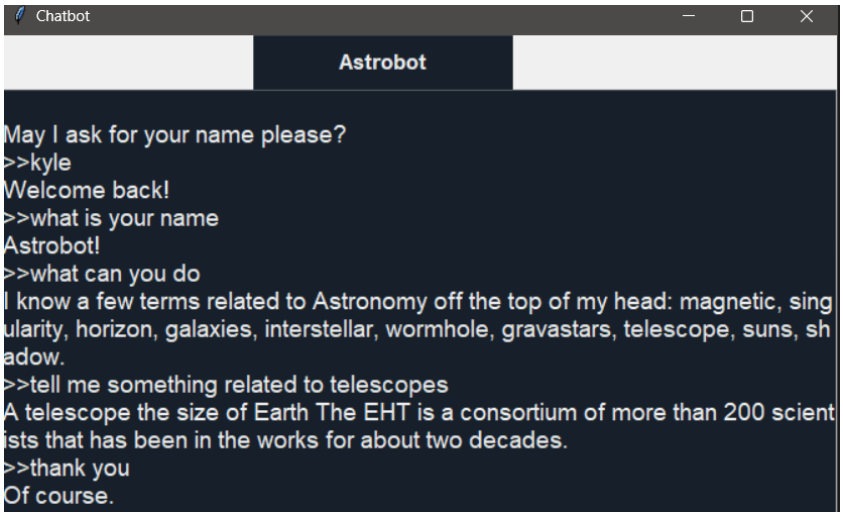
Along the way, we are also recording determining whether what the user is saying is considered a positive, negative, or objective statement. When a user asserts a statement that is positive, we add that to the user model as a topic of interest. When a user asserts a statement that is negative, we remove that from the user model as a topic of interest if it existed there previously. If it is objective, we continue on without making any changes. We determine this by using the natural language processing technique of sentiment analysis. The specific path we went about this with is a process known as VADER (Valence Aware Dictionary and sEntiment Reasoner).

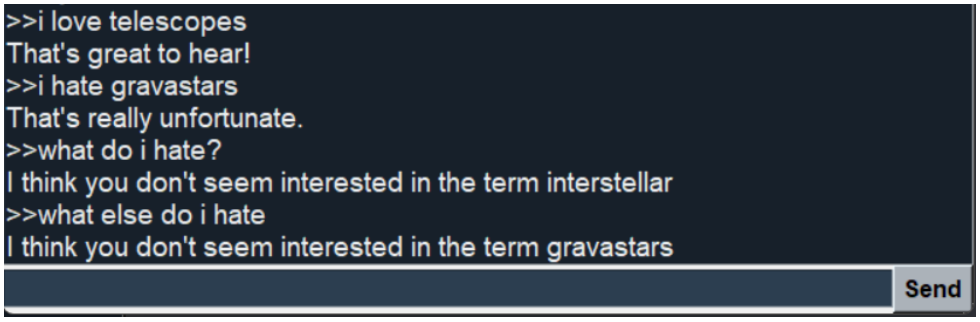
Our knowledge base was developed using a web crawler we had previously created in a prior project. The web crawler extracts contents from different web pages which are relevant to black holes and astronomy. The top 50 words were chosen from those web pages using the NLP technique known as TF-IDF (Term Frequency - Inverse Document Frequency). From this list we handpicked the top 10 words we found the most relevant. These words are magnetic, singularity, horizon, galaxies, interstellar, wormhole, gravastars, telescope, suns, and shadow. These are 10 central topics that our chatbot is oriented around conversing with. The knowledge base is stored in a pickled dictionary, with each of these key topics associated with a list of relevant responses on the subject. See the appendix on the knowledge base for a sample of one of the topics.

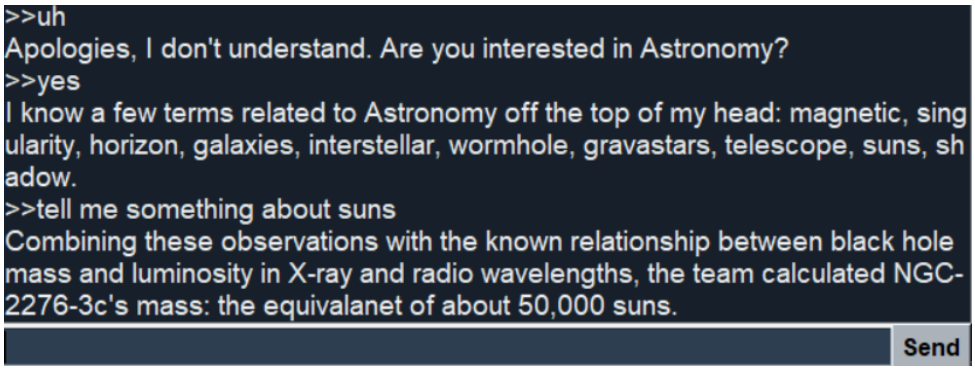
Evaluation

We developed the chatbot from scratch without using any prior framework, which makes the system very simple and prone to error. There are some problems in the system when it comes to asking very detailed questions that go beyond simply asking for an explanation of a certain topic. For example, the bot could have a difficult time differentiating the question “What is a telescope used for” and “How is a telescope built”. The system will see that a question about a telescope is being asked, so it will enter our knowledge base for an answer. It would then answer the first question very accurately but struggle with the second question. This is both that we may not have the answer in our knowledge base as this is not a question about astronomy and the bot is not sophisticated enough to recognize that.

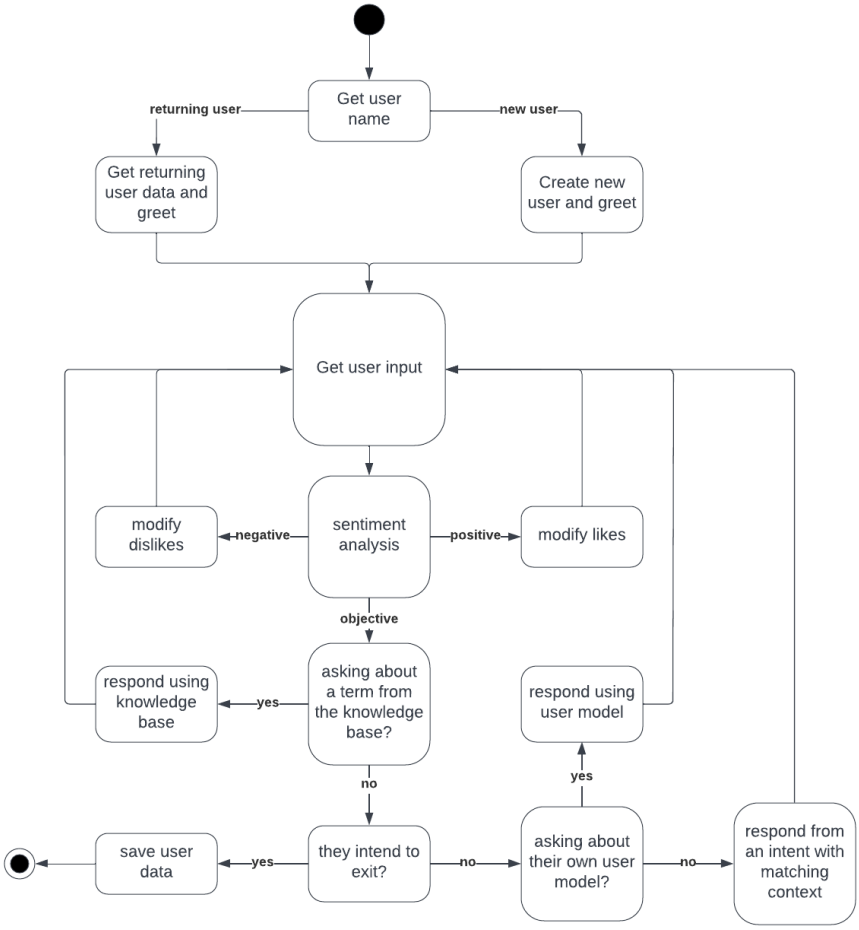
Sample Dialog Interactions







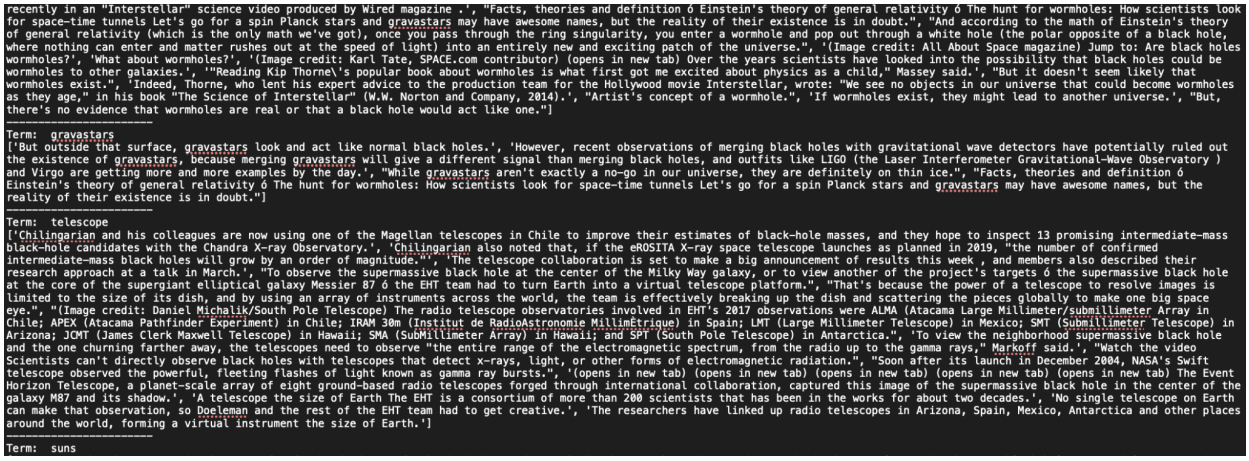
Dialog Logic



Appendix User Model

```python
import pickle

class User:
    def __init__(self, name, likes, dislikes):
        self.name = name
        self.likes = likes
        self.dislikes = dislikes

user1 = User("john", ['planet', 'galaxy', 'star'], ['blackhole', 'magnetic'])
user2 = User("kyle", ['blackhole', 'magnetic'],['interstellar', 'wormhole'])
user3 = User("ryan", ['interstellar', 'wormhole'], ['singularity', 'galaxy'])

user_list = [user1, user2, user3]

pickle.dump(user_list, open('user_list.p', 'wb'))
```

Appendix Knowledge Base



The above photos are samples of parts of our knowledge base. Each term value has a list of sentences that are associated with it. These sentences were gathered using a web crawler that scraped information on relevant sites associated with black holes. The terms themselves were determined using TF-IDF (Term Frequency - Inverse Document Frequency) to see what terms were most relevant within the text that we had gathered.