Christian Kavouras
1/30/2021
HW 2, Q6

In implementing the Dialog Manager in this portion of the project, we explored the challenges involved in maintaining state, managing interaction with a database, and balancing conversation based on various levels of initiative. With respect to the FST based DM implemented in Q1/Q2, we were able to relatively quickly get a working prototype for gathering basic information from the user. This conversation style felt like an old fashioned customer service agent, in which the user did not have much room to take initiative. As we moved toward a mixed-initiative model in Q3/Q4, we were able to handle the user talking “out of turn” (i.e. giving input that isn’t asked for, or updating a previous piece of information). The mixed initiative model does a good job of allowing the user to feel more comfortable to speak their mind. It allows the user to give multiple levels of input at once and decide to change their mind about previous decisions. Additionally, we added some support for grounding statements to balance the user’s requests, and to acknowledge confirm/deny statements.

Implementing a mixed initiative system was the most rigorous and time consuming portion of our work, and required a fair amount of forethought and reworking of the system design as we went along. The initial development happened in chunks, adding small pieces of functionality until we gained a better understanding of the needs of the system (for example, what type of information to store in the Dialog Frame as opposed to the Semantic Frame, and when to interact with a database). As a result of this, much of the logic in the Dialog Manager became bloated with conditional statements that could become hard to understand. As we improve upon this, we will hopefully refactor, for example, logic for selecting a dialog act by streamlining the expected states, flags, etc, that the DialogFrame takes on and more elegantly anticipating the flow of its state. As it stands the Dialog Manager and Dialog Frame are not abstracted as well as they could be - our Dialog Frame is essentially a property of the Dialog Manager, which makes them feel like one in the same structure. The Semantic Frame, by contrast, is just a messenger that updates the Dialog Frame on the state of the user. We wanted to avoid building on the Semantic Frame’s structure too deeply, instead leveraging two variables, Intent and Slots, that send information to the DM. The “slots” are a dictionary that can really contain any sort of information, which kept this implementation flexible.

One interesting portion of building out the Dialog Frame was implementing some grounding and acknowledgement. In order to increase its fluidity, we added some system flags to keep track of what new information was added last, if any. If new information was added, we ground, using a varied distribution of multiple different grounding statements. I am curious to see how future (machine learning oriented) versions of the NLU can determine how informative a user is on a given turn, and whether to ground.

Regarding the NLU and NLG components of our system - while these are still not developed deeply, we were able to do a fair amount of use case coverage with just key word recognition (using regex) and smart logic handling.