Tracy Rohlin, Travis Nguyen

Prof. Gina-Anne Levow

LING 575 (Spoken Dialog Systems)

June 7, 2017

Project Report

Our project is a mixed initiative task-oriented spoken dialogue system focused on kitchen tasks. The project was inspired by common questions that Tracy frequently has to type into her laptop while cooking dinner, such as "How much juice is in a lime?" or "What can I substitute for sour cream?" We chose to develop our system using the Alexa developer console provided by Amazon, partly due to having the Amazon Echo (and thus understanding its limited capabilities) already, and partly due to Tracy's desire to become familiar with the developer console before she begins her internship at Amazon.

Task-oriented spoken dialogue systems is a rich field that has been studied and has led to a plethora of implementations. For example, Litman, Young, Gales, Knill, Ottewell, van Dalen, and Vandyke (2016) have developed three separate corpora for the purpose of studying the English skills of non-native users. These corpora have been designed around task-oriented spoken dialogue systems: one where the user is tasked to find bus routes in Pittsburgh, another where the user must find Michelin star restaurants with a specific set of criteria, and one where users are instructed to find laptops with particular specifications. In other research, Martins, Pardal, Franqueira, Arez, and Mamede develop a spoken dialogue tutoring system based on DIGA (DIaloG Assistant) that teaches the user how to cook a recipe. The DIGA system was originally developed as a domain-independent framework that eventually became the basis for a

smart-home butler application and an informational database application. Additional research around task- and home-oriented applications include an interactive TV entertainment system (Ibrahim and Johansson, 2003) and a system that logs and calculates nutritional data for users struggling with obesity (Korpusik and Glass, 2017).

In order to implement such a kitchen helper system, an informal poll was taken among target users on what they would like to see in such a spoken dialogue system. The responses included conversion functions, for both converting within the imperial system (e.g., gallons to cups) and for converting between metric and imperial. In addition, potential users desired a function that would indicate when a dish was done (e.g., quick bread, cakes, etc.) or how long a piece of meat would take to roast or cook to the correct internal temperature. Survey participants also wished for common substitutions and for the Alexa to read recipes to them. This last task was not undertaken by our system due to feasibility constraints. In particular, it is difficult to determine what recipe a user requests. For example, Epicurious, has 71 variations for "scalloped potatoes," each with their own list of ingredients. As such, we focused mainly on conversions, substitutions, and calculating cooking times.

The project was constructed in Python using both the Flask and Flask-Ask libraries. The Python code itself is relatively simple, constructed either using simple mathematical formulas or dictionary look-ups. For example, the code to calculate roasting times contains a large dictionary of different meats, each with a minimum and maximum roasting time per pound of meat. The user typically indicates how many pounds they would like to cook, and if not provided, the default of 1 lb is used. Afterwards, a simple multiplication provides the time range for roasting

that cut of meat. Other examples of functionality include converting temperature between Celsius and Fahrenheit, and converting between kilograms and pounds in terms of weight.

The challenge came with converting user responses into precise numerical value. The str_to_dec() and dec_to_str() functions convert from string to float and from float to string, respectively. The str_to_dec() function only recognizes a handful of fractions (such as "one-half", "one-eighth"), so it has limited capabilities recognizing non-standard (by way of cooking) fractions. Likewise, the dec_to_str() function will convert common fractions such as ½ and ¾ into expanded words, but uncommon fractions are left as truncated decimals. One challenge that we encountered with this is that the Alexa will convert some fractions to string representations, such that "one eighth" becomes "⅛", as opposed to leaving them as string literals. To alleviate conversion errors, our program searches for and splits on the basis of the unicode character "u2044" (i.e., "/"), converts the two constituents into floating-point numbers and performs the necessary division. Lastly, the speak_decimals() function converts mixed numbers into more natural speech-like strings such that "2.5" is converted to "two and a half."

Other challenges faced were related to the Amazon developer console itself. Tracy found several tutorials on how to host code using AWS and link the code to the Alexa skill on Amazon's developer console. Unfortunately, neither of these tutorials worked and the console was too much of a "black box" to properly debug. Instead, Travis found a tutorial using ngrok to host the code and link it to the Alexa skill. Every time ngrok is run, however, it creates a different URL endpoint with which to connect to; stable URL endpoints are only provided under the paid plan. As such, every time ngrok is run the configuration must reset using the new URL. The advantage of using Flask and ngrok, however, is that any code changes or bug fixes can be

made while the code is still running; Flask will then recognize the changes to the code without having to exit out both ngrok and the python code, and thus the session is not interrupted.

Another challenge faced was due to the nature of an Alexa skill itself. Compared to other more flexible systems like VoiceXML, the Alexa Skills Kit requires several utterances if slots are made optional. For example, if a specific function has the parameters “whole_num”, “frac_num” and “meat_type”, with default values for “whole_num” and “frac_num”, three separate utterances must be created: One containing just {whole_num} as a slot variable, one containing {frac_num}, and one containing both {whole_num} and {frac_num}. We also were required to make several dummy slots so that multiple verbs could be used within an utterance, such as “is” and “are” or “would like” and “want.” This was done to avoid making utterances for each type of verb or modal and creating too long a list of utterances. In addition, our Python code has a specific list of values recognized for each function and while some mappings exist (e.g. “butter” is mapped to “salted butter”), strings returned by Alexa’s speech recognition program that do not exactly match our dictionaries’ keys will not produce the desired result.

Despite these challenges, Alexa successfully recognized our utterances and returned the correct result from our Python functions when the system was tested by both members of our group. Each function was tested using the built in TTS testing module on the developer console, and several were tested using EchoSim.io’s Alexa simulation page. Had we more time, it would have been interesting, albeit challenging, to implement the recipe reading function such that perhaps the Alexa would give a list of websites or recipes to choose from before reading the selected one. Although this was never implemented, we are satisfied with both the functionality and speech recognition of our project as is.

References

Common Ingredient Substitutions (Infographic) - Allrecipes Dish. (2017). *Allrecipes Dish*. Retrieved 15 May 2017, from http://dish.allrecipes.com/common-ingredient-substitutions/

Ibrahim, A., & Johansson, P. Multimodal dialogue systems for interactive TV applications. *Proceedings. Fourth IEEE International Conference On Multimodal Interfaces*. http://dx.doi.org/10.1109/icmi.2002.1166979

Korpusik, M., & Glass, J. (2017). Spoken Language Understanding in a Nutrition Dialogue System. *IEEE/ACM Transactions On Audio, Speech, And Language Processing*, 1-1. http://dx.doi.org/10.1109/taslp.2017.2694699

Litman, D., Young,, S., Gales,, M., Knill, K., Ottewell, K., van Dalen, R., & Vandyke, D. (2016). Towards Using Conversations with Spoken Dialogue Systems in the Automated Assessment of Non-Native Speakers of English. In *SIGDIAL 2016 Conference* (pp. 270-275). Los Angeles: Association for Computational Linguistics.

Martins, F., Pardal, J., Franqueira, L., Arez, P., & Mamede, N. (2008). Starting to cook a tutoring dialogue system. *2008 IEEE Spoken Language Technology Workshop*. http://dx.doi.org/10.1109/slt.2008.4777861

Meat and Poultry Roasting Chart. (2017). *Foodsafety.gov*. Retrieved 15 May 2017, from https://www.foodsafety.gov/keep/charts/meatchart.html

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T., & Hetherington, L. (2000). JUPlTER: a telephone-based conversational interface for weather information. *IEEE Transactions On Speech And Audio Processing*, *8*(1), 85-96. http://dx.doi.org/10.1109/89.817460