SELF- ANNOTATION THROUGH CONVERSATIONAL GROUNDING

by

Aprajita Shukla

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Computer Science

Boise State University

August 2018

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Aprajita Shukla

Thesis Title: Self- Annotation Through Conversational Grounding

Date of Final Oral Examination: 21st August 2018

The following individuals read and discussed the thesis submitted by student Aprajita Shukla, and they evaluated the presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Dr. Casey Kennington, Ph.D.	Chair, Supervisory Committee
Dr. Jerry Alan Fails, Ph.D.	Member, Supervisory Committee
Dr. Maria Soledad Pera, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Dr. Casey Kennington, Ph.D., Chair of the Supervisory Committee. The thesis was approved for the Graduate College by Tammi Vacha-Haase, Ph.D., Dean of the Graduate College.

Dedicated to my parents who are highly statistically significant.

ACKNOWLEDGMENTS

I wish to express gratitude to the Computer Science Department at Boise State University for funding this project, my advisor - Dr. Kennington and SLIM our research group for their constant support, guidance and feedback.

I would like to thank my family and friends for bearing with me when I missed out on important events to work towards completing my degree.

I would also like to thank Coursera and Udemy for having courses that helped me understand the material I needed the most while working on my thesis.

And last but not least, I would like to thank the committee members for agreeing to be on my committee.

ABSTRACT

An *abstract* is a brief summary of the document. A typical abstract provides a brief introduction, enough to provide context for the document, explains the purpose of the thesis or project, and summarizes the major results and conclusions. Keep in mind that a casual observer is likely to judge the content of the document by the abstract and title alone. (There is an old addage: "in a joke, the punchline comes at the end; in a paper [or thesis], it comes in the abstract.") A single concise paragraph usually suffices for the abstract. If it spills onto a second page, it is probably too long.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

LOL – Laughing Out Loud

OMG – Oh My God! now is the time for all good men to come to the aid of their country.

LIST OF SYMBOLS

$\sqrt{2}$ square root of 2

λ lambda symbol, normally used in lambda calculus but it sometimes gets used for wavelength as well

CHAPTER 1

INTRODUCTION

A Dialogue System (DS) is a program that interacts with a human in a natural language, usually through text, speech (or both). Prevalent DSs are commonly used by people from all walks of life to perform small and easy tasks like getting weather updates, setting up a reminder or making a call to a phone contact. Generally speaking, such tasks are performed by one's personal assistant; therefore, it is befitting to collectively refer to them as Digital Personal Assistants (DPAs) e.g. SIRI, Alexa, Cortana, Foxsy, Meekan, just to name a few. Based on the mode of communication, these DPAs can be categorized mainly into (a) Spoken Dialogue Systems (SDSs) that use speech to interact with the user, e.g. Cortana (b) Chatbots that use text, e.g. ELIZA and (c) Multimodal Dialogue Systems (MDSs) that can process two or more user inputs together, e.g. MATCH. Our thesis work uses a chatbot interface but is applicable to all the types.

Irrespective of the communication medium, interaction with all the DPAs happens through language use and takes the form of dialogue. By virtue of being a human we are accustomed to a certain way of engaging in dialogue. Since humans use DPAs to accomplish tasks, delegate small jobs and enable multi-tasking in a hands-free situation, the nature of conversing with them should, to the greatest extent possible, emulate a conversation between two human participants so as to reduce cognitive load

on the human and foster a dialogue that is effortless and less frustrating, and feels more natural. Through our research, we strive to improve that experience to provide a conducive environment for increasing society's DS usage in the future.

Improvements can be done on two fronts: (a) Improving the level of technology, which enhances the range of applications or activities that a DPA can help accomplish and (b) Improvement in the User Experience (UX), which improves the ease with which the interaction can take place, ensuring that the user wants to interact with it again and again. A case in point is that we come across DPAs that support the use of different interfaces for ease of interaction resulting in greater appeal to a wider user base. Due to rapid advancements on both fronts, we require a constant effort to make sure that the DPAs keep up. Therefore, this work focuses on implementing *conversational grounding* through incremental processing in existing DPAs that targets the important aspect of naturalness and provides the basis for future advancements.

The process of establishing *common ground* in dialogue is called *grounding*. In any dialogue, participants take turns to communicate successfully. Dialogue between two participants is a type of joint activity made up of joint actions called dialogue acts, e.g. utterances, clarifications, feedback, etc. A joint action is one performed by an ensemble of people acting in coordination with each other. To complete any joint activity, participants need to perform joint actions to advance and they cannot take joint actions without assuming certain pieces of common ground [3]. Since dialogue participants take actions as a whole, they need to have a combined understanding of all the fundamental knowledge that is prerequisite to performing a joint action. This shared understanding is based on each participant's beliefs, knowledge, culture, perceptions of the world, etc. This shared understanding is called common ground and it has three components. First is the knowledge that each participant has before the

start of the joint activity; second is the knowledge they gain with each completed step of the joint action; third is the..... So the common ground is first established and then information is added to it with each step incrementally to include the common knowledge they share about the task as it proceeds. Some information is also removed from the *common ground* using clarifications. The remarkable thing about common ground is that both participants are aware of what each of them know and also know what the other participant knows. This common ground is the key to coordination in dialogue.

Ideally, a DPA should attempt to establish common ground between itself and users by signaling understanding and by learning a mapping between the words that the users actually express and the system actions [7]. Showing this internal representation to the user lets the user know and judge the level of understanding of the system. It also is a sort of feedback to the user to let them know that the system has an understanding with the user. When this process is repeated, with every dialogue act, system and user can add to that shared common ground and draw from it if and when required for future dialogue.

It is long established that common ground is key to a successful dialogue. A significant body of work has investigated common ground in human-human communication, for example, how the shared culture background [2] and spatial reasoning capabilities affect human communication. This is also true when the participants of a joint action are a system (personal assistant, a dialogue system or a chatbot) and a human being. What makes communication among humans successful is the common ground we have: mutual beliefs and knowledge about the shared environment–about what both can see, about what both have already talked about. Common ground is also a form of self-awareness. For example, two people, Susan and Bill, are aware

of certain information they each have. To be common ground, their awareness must be reflexive - it must include that very awareness itself [3]. In situated human-robot dialogue, although humans and robots are co-present in a shared environment, they have significantly mismatched capabilities in perceiving the shared environment. In order for humans and robots to communicate with each other successfully using language, it is important for them to mediate such differences and to establish a common ground [2].

In today's PAs, we see this common ground is not well established before the required action is performed. The user usually assumes that the system understood a user utterance because it did not ask for any clarifications. Current systems only display ongoing understanding of an utterance by displaying the state of the transcribed speech to the user, but even perfect transcription does not mean understanding [6]. What is not displayed is the system's semantic representation of the transcription because it's not easily understandable by users. Sometimes the user might see an"okay" or see a"yes" pop up on the screen when a chatbot tries to signal understanding. This "okay" or "yes", however, does not necessarily mean that it understood the user; it usually is just the system recognized transcript and there is no guarantee that the request was actually understood and could lead to the wrong system action.

Systems that use data-driven statistical models tend to work more robustly than systems that are rule-based. These data-driven systems require large amounts of annotated data to learn the mappings between spoken input and system response. Getting this data requires a lot of time, effort and money. This is a huge problem in

the data science field because to test a new system one should have relevant data–and fast. Training data is tied to grounding in an important way: if system models learn how to map between user utterance and system response, then that very mapping is a learning of what kinds of actions are expected. This is an important insight because, for the systems in consideration, the best way to collect data is by conducting user studies which again take time and require most of the parts of the system to be functional before testing it with real users, but the nature of the collected data might be different from how it will actually be used. The problem with this is that user studies can not be conducted in the experimental stage; i.e. when most of the parts of the project are not properly working. So, whenever such systems need to be trained, its creators have to look for a lot of data in order to improve their initial models. It is a good idea if one already has datasets available for the kind of system they set out to make, but for systems that are going to work on new data, initial lack of it, is a huge roadblock in making marked progress.

There is one more shortcoming of current spoken dialogue systems that is tied to grounding: PAs and SDSs lack long-term memory. Users get easily frustrated when the system does not remember doing something that happened only a moment prior and sometimes this is a huge reason why people don't like interacting with PAs. The following interactions between a known PA system (s) and a user (U) illustrates this [9]:

(1)

a. U: Hey S, call my mom.

b. S: I don't know "mom."

c. U: My mom is Martha.

d. S: OK, calling Martha.

(2)

a. U: Hey S, call my mom.

b. S: I don't know "mom."

By uttering OK, the system makes the user think that system signaled understanding. However, the user was later surprised that the system misunderstood, as evidenced in (2-a). This example also points towards the grounding problem between the user and the system [2]. More specifically in this case the system had no memory to store and recall facts about users and interactions in order to build mutual understanding. For this kind of grounding to be accomplished, system should dynamically update with each interaction while signaling understanding to the user, following [9].

As a recap, the three shortcomings of DPAs which I plan to remedy are the following:

- Systems show a lack of establishing and building common ground (i.e., grounding)
- Systems require large amounts of training data to work robustly
- Systems have no long-term memory

1

[1]Too many footnotes, however, can be distracting.

CHAPTER 2

THESIS STATEMENT

As a solution to the above issues, following my previous work in [7], I will leverage a Graphical User Interface (GUI) to display the internal state of the system in an intuitively readable way to the user and construct a system that works without providing it with any training data and have it learn autonomously as it interacts with the user. I will apply my model in a domain where long-term, episodic memory is a crucial part of the system's functionality: a chatbot that builds contact lists. I explain in greater detail below how I will accomplish these goals.

Crucial to the success of my thesis is *how* the SDS is modeled. One reason why today's systems have the above issues (grounding, data, and memory) is because they do not process *incrementally*. That is, today's systems usually need to be invoked with some kind of wake word, then the system waits until a minimal amount of silence before processing. An incremental SDSs, like the one I explain in greater detail below, does not wait before it starts to process; it processes as much as possible as early as possible, usually word by word. An incremental system can immediately display feedback, misunderstandings, and request clarifications (which are important to grounding) as a user's utterance unfolds allowing the user to track the system's understanding in real-time. With these signals, a user can fix any errors the system

makes quickly and with minimal effort, which gives the system positive and negative examples of how it should respond to user input, effectively providing the system with training data as it interacts (yet another form of grounding). Finally, the application domain of an incremental chatbot (a type of dialogue system where the input medium is text instead of speech), purposefully gives the system information that is important to remember (which is also tied to grounding) where the chatbot responds to input as the user types instead of waiting for a user to submit an entire request at once.

Incremental input will be an important component for facilitating grounding, which will be realized in the thesis through the graphical user interface (GUI). The idea is to make the user label data as and when they are entering it. In this way, the user will have the freedom to add, correct and confirm their input at the same time while the processing happens along the way. This will improve the accuracy and speed of processing, not to mention the naturalness that will be added to the interaction, making it resemble a human-human interaction to a higher degree than it does currently.

Once the grounding is in place, we want to focus on finding out how the memory improves with it. Our expectation is that it gets better with grounding. Moreover, we are dealing with incremental input so we will need to have a way the system stores incremental units of the utterance till it is complete.

CHAPTER 3

BACKGROUND AND RELATED INFORMATION

Common ground isn't just there, ready to be exploited; people have to establish it with each person they interact. The first step in establishing either type of common ground is finding the right shared bases–the right evidence. E.g. If a father and a son were at the beach, their joint gaze on a conch shell would be excellent evidence that each of them have that information that there is a conch shell between them but it is poor evidence that they each have the information that the shell is six years old [3] *page 98*. The father could judge it highly likely that the conch shell is part of our common ground, but unlikely that its age is [2]. Shared bases vary in what H. H. Clark calls quality of evidence and the type of information they give rise to. People are fallible in these judgments and they know it. Fortunately, we have practical strategies in using language for preventing such discrepancies and repairing them when they arise[3] e.g asking each other questions to remove certain assumptions we had before the dialogue started but later on turned out to be false. Therefore, when it comes to coordinating on a joint action, people can not rely on just any information they have about each other. They must establish just the right piece of common ground, and that depends on them finding a shared bases for that piece. The shared basis is what Schelling called the key to the coordination problem and what Lewis called the coordination device.[2]

It takes two people working together to perform joint activities e.g., play a duet, shake hands, play chess, waltz or teach. To succeed, the two of them have to coordinate both the content and process of what they are doing. Alan and Barbara on the piano, must come to play the same Mozart duet. This is coordination of content. They must also synchronize their entrances and exits, coordinate how loudly to play forte and pianissimo, and otherwise adjust to each other's tempo and dynamics. This is coordination of process. They cannot even begin to coordinate on content without assuming a vast amount of shared information or common ground-that is, mutual knowledge, mutual beliefs, and mutual assumptions.[2] [4] [8]. And to coordinate on process, they need to update their common ground moment by moment. All collective actions are built on common ground.

What does it take to contribute to conversation? For example, suppose Alan utters to Barbara, "Do you and your husband have a car?" In the standard view of the speech acts[1], what Alan has done is ask Barbara whether she and her husband have a car, and, in this way, he has carried the conversation forward. But this isn't quite right. Consider this actual exchange:

Alan: Now,-um, do you and your husband have a j- car

Barbara: - have a car?

Alan: Yeah

Barbara: No -

Even though Alan has uttered "Do you and your husband have a car?", he hasn't managed to ask Barbara whether she and her husband have a car. We know this

because Barbara indicated with "-have a car?", that she hasn't understood him (Actually, the word ask is ambiguous between "utter an interrogative sentence" and "succeed in getting the addressee to recognize that you want certain information." You can say, " Ken asked Julia, 'Are you coming?' but failed to ask her whether she was coming because she could not hear him." We are using ask in the second sense here.) Only after Alan has answered her query (with "yeah") and she is willing to answer the original question("no-") do the two of them apparently believe he has succeeded. So asking a question requires more than uttering an interrogative sentence. It must also be established that the respondent has understood what the questioner meant.

Looking at these above observations, we can see that this is not true for human-robot dialogue since Chatbots, PAs, SDSs do not share the same world representation as us humans. Their representations of the shared world are misaligned with that of human beings. Since their design is limited to just one channel of communication i.e. the verbal channel or in some cases text, they do not possess the same number of channels that we humans do to perceive information. Extracting information from what we perceive through our senses is very normal for us, but not an option for robots.

Usually participants add to common ground incrementally i.e. as and when one person speaks and the other understands and agrees. A lot of psycholinguistic research has shown that humans don't process each sentence as and when they hear it, but they understand it piece by piece and answer only after waiting for the last utterance to be heard. In PAs etc. incremental processing is uncommon which makes our

interaction with them less natural and not appealing to the users.

In [2] we can see a dialogue system that attempts to mediate a shared perceptual basis between the human and the robot through automatic knowledge acquisition. As conversation proceeds, the robot first matches human descriptions to its internal representation of the shared world. It then automatically acquires and confirms through dialogue common ground knowledge about the shared environment. The acquired knowledge is used to enrich the robot's representation of the shared world. Their results have shown that an extra effort from the robot, to make its human partner aware of its internal representation of the shared world, contributes to better common ground. This was applied to a setting involving a human and the NAO robot. This kind of work is yet to be accomplished in situations where the interaction is between a chatbot or an SDS and a human.

Similar to our work, Julian Hough and David Schlangen presented a simple real-time, real-world grounding framework in [5], and a system which implements it in a simple robot, allowing investigation into different grounding strategies but their emphasis was on the grounding effects of non-linguistic task-related actions. They experimented with a trade-off between the fluidity of the grounding mechanism with the 'safety' of ensuring task success. They had a simple pick-up-and-place robot with uni-modal communication abilities, which is simply its manipulation behaviour of objects. They used basic reinforcement learning to make the robot realize when it made a mistake, so, they had a mechanism which deducted some points for a mistake. This ensured that the robot at least asks for a clarification, the next time it is not sure of what to do next. Likewise, a positive reinforcement mechanism worked when

it did the right thing and promoted it to do the right thing the next time it thought so. While this robot did not have natural language generation (NLG) capabilities, its physical actions are first class citizens of the dialogue so it is capable of dialogic behaviour through action.

My previous work, a system called amBrOISEa [7] tries to establish and build common ground between itself and users by signaling understanding and by learning a mapping via interaction between the words that users actually speak and the system action. The system generates clarification request once the utterance is complete. The user then chooses 'yes' or 'no' to let the system know if the interaction was correctly understood. Those interactions that went well (i.e. the system understood correct intent) are then added as training data from which the model later learns.

My thesis builds directly off of the amBrOISEa model. We go beyond that previous work by improving the GUI which we use to signal system understanding to the user as the user types and by applying the incremental model of natural language understanding to a domain where remembering important facts is the primary purpose of the system. For storing and retrieving information, we build off of [9] that applied a graph database to represent structured information. The difficulty lies in mapping from a surface form utterance (i.e., a sequence of words) to isolate the important bits of information that need to be stored. Though we attempted to build a system that can work with minimal training data in previous work [7], the system was only able to interact with users for a couple of minutes and anything that was learned was then forgotten. For my thesis, I will set up a system that continues to learn and improve over time–a system that not only remembers facts, but learns how to better interact

as it interacts.

This holds not only for interpersonal dialogue, but also when one of the participants in the dialogue is a machine. The introduction of common ground facilitates *conversational grounding* which is very important.

CHAPTER 4

OUR MODEL OF CONVERSATIONAL GROUNDING

Our proposed method lends itself to conversational grounding (i.e., signaling ongoing understanding) to know if the system and the user are able to understand each other and create a shared common ground of knowledge to keep adding to as the conversation unfolds. Applied to any DPA, SDS, or chatbot, we can bootstrap a system to work reasonably well with minimal or no training data, and, using GUI to make up for the shortcomings of the system, we can signal understanding in an intuitive, graphical way and allow the user to correct system mistakes explicitly, while only minimally disrupting the interaction between the system and user. In this way, users annotate their input directly though it seems as if it's just a normal dialogue.

4.0.1 Natural Language Understanding

In amBrOISEa [7] we approached the task of Natural Language Understanding (NLU) as a slot-filling task[6], where an intent is complete when all slots of a frame are filled. The main driver of the NLU will be the SIUM model introduced in Kennington et al. (2013). SIUM has been used in several systems which have reported substantial results in various domains, languages, and tasks[6]. Following Kennington and Schlangen (2016), though originally a model of reference resolution, it was always intended to

be used for general NLU. The model is formalized as follows:

$$P(I|U) = \frac{1}{P(U)} * P(I) \sum_{r \epsilon R} P(U|R = r) P(R = r|I) \tag{4.1}$$

That is, P(I—U) is the probability of the intent I(i.e., a frame slot) behind the speaker's (ongoing) utterance U. This is recovered using the mediating variable R, a set of properties which map between aspects of U and aspects of I. We used this model in [7] to get us started with no training data. We opted for abstract properties (e.g., the frame for restaurant might be filled by a certain type of cuisine intent such as Italian which has properties like pasta, Mediterranean, vegetarian, etc. Properties were pre-defined by a system designer and can match words that might be uttered to describe the intent in question. For P(R—I), probability is distributed uniformly over all properties that a given intent is specified to have. (If other information is available, more informative priors could be used as well.) The mapping between properties and aspects of U can be learned from data. During application, R is marginalized over, resulting in a distribution over possible intents. This occurs at each word increment, where the distribution from the previous increment is combined via P(I), keeping track of the distribution over time.

4.0.2 Graphical User Interface

The goal of the GUI is to intuitively inform the user about the internal state of the ongoing understanding. One motivation behind this is that the user can determine if the system understood that user's intent before providing the user with a response (e.g., a list of restaurants of a certain type); i.e if any misunderstanding takes place,

it happens before the system commits to an action and is potentially more easily repaired, incrementally. the display in [7] was a right branching tree, where the branches directly off the root node display the affordances of the system (i.e., what domains of things it can understand and do something about). When the first tree is displayed, it represents a state of the NLU where none of the slots are filled.

When a user verbally selects a domain to ask about, the tree is adjusted to make that domain the only one displayed and the slots that are required for that domain are shown as branches. The user can then fill those slots (i.e., branches) by uttering the item to fill the slot directly. For, example, at a minimum, the user could utter the name of the domain then an item for each slot directly, (e.g. food Thai downtown) or the speech could be more natural(e.g., I'm quite hungry, I am looking for some Thai food maybe in the downtown area). Crucially, the user can also hesitate within and between chunks, as advancement is not triggered by silence thresholding, but rather semantically. When something uttered falls in to the request state of the DM as explained later, the display expands the subtree under question and marks the item with a question mark. At this point, the user can utter any kind of confirmation. A positive confirmation fills the slot with the item in question. A negative confirmation retracts the question, but leaves the branch expanded. the expanded branches are displayed according to the their rank as given by the NLU's probability distribution. Though a branch in the display can theoretically display an unlimited number of children, we opted to show only seven children; if a branch had more, the final child displayed as an ellipsis.

A completed branch is collapsed, visually marking its corresponding slot as filled.

At any time, a user can backtrack by saying no (or equivalent) or start the entire interaction over from the beginning with a keyword, e.g., restart. To aid the user's attention, the node under question is marked in red, where completed slots are represented by outlined nodes, and filled nodes represent candidates for the current slot in question.

4.0.3 Incremental Processing

Dialogue processing by its very nature is *incremental* [10]. No dialogue agent (artificial or natural) processes whole dialogues, if only for the simple reason that dialogues are *created* incrementally, by participants taking turns. At this level, most current implemented dialogue systems are incremental: they process user utterances as a whole and produce their response utterances as a whole.

Incremental processing, as the term is commonly used, means more than this, however, namely that processing starts before the input is complete. Incremental systems hence are those where "Each processing component will be triggered into activity by a minimal amount of characteristic input." If we assume that the characteristic input of a dialogue system is the utterance, we would expect an incremental system to work on units smaller than utterances.[5]

The results of [7] were positive overall: the system is useful and allows users to fill an itinerary using speech. Users were able to recreate their itineraries with the autonomous system much more accurately than with the baseline system. Minimal grounding indeed took place through the GUI by the tree and map, both of which

updated incrementally as the users' utterances unfolded, by properties (i.e. ontology) discovery by the system, and by improving the mapping between utterances and properties. This will be a starting point for this project to leverage amBrOISEa or similar kinds of systems to autonomously improve the dialogue manager and the NLU. This will involve a strong influence of grounding which will in turn, help in making the system understanding better.

For grounding we will bootstrap this system by overcoming the shortcomings of the untrained NLU by showing buttons and clarification requests in the GUI, allowing the user to make the system understand intent in real-time and be able to make changes to it if required.

SDS takes the SIUM code from [7] which has the working components mentioned above and add that to the domain of a chatbot that interacts with the user while they type in utterances. The system will ask the user to correct errors, as and when they are trying to enter data, this way, the labeling will happen along with the dialogue. It makes the experience of communication more natural and adheres to the principle of least collaborative effort. People apparently don't like to work any harder than they have to, and in language this truism has been embodied in several principles of least effort. (Grice 1975) expressed this idea in terms of two maxims: Quantity - make your contribution as informative as required for the current purpose pf the exchange, but do not make your contribution more informative than is required - and Manner - Be brief(avoid unnecessary prolixity). According to this version, speakers are supposed to create what we will call proper utterances, one they believe will be readily and fully understood by their addressees.

CHAPTER 5

DATA COLLECTION

5.0.1 Amazon Mechanical Turk

We used Amazon's Mechanical Turk (MTurk) to collect data for Data Analysis. It is a marketplace for work that requires human intelligence. The web service enables a company or an individual to programmatically access this marketplace and a diverse, on-demand workforce. Developers can leverage this service to build human intelligence directly into their applications.

usually collecting data is not not Since data collection is a tedious job, we do not need to make the same in the

5.0.2 Database

5.0.3 Data Analysis

CHAPTER 6

EXPERIMENT AND EVALUATION

6.0.1 Experiment

Task n Metric (people domain, no of clarifications, of happy faces, restarts, go backs goes down) Implementation details Used Inprotk, SIUM, heroku, etc. How did the model improve over time? Hypothesis: model improves as time goes by: Number of clarifications goes down Number of happy faces goes up Number of restarts goes down Number of go backs goes down Number of frowny faces goes down

Task

we asked each user to imagine themselves to be the career center head at a prestigious university attending a Gala to network with people to be able to invite them to participate in their university wide Career Fair at a later date. The aim was to use the app and try to enter basic contact information for as many people as possible. The basic information could include some or all of these categories in any order: *First Name*, *Last Name*, *Email*, *Web Address*, *Work* and *Notes*.

6.0.2 Evaluation

As our system will be online over the course of several weeks for many people to interact with, it should ideally be improving its ability to interact while remembering

facts that individual users gave to it. We therefore wish to capture the user experience at each use. To accomplish this without interrupting the interaction between the users and our system, we will again leverage the GUI: after a short dialogue where the system inputs facts into its database (which could have been preceded by clarifications and repairs by the user), a simple happy face icon and a sad face icon will display. A user then has the option of clicking or tapping on one of these icons to determine if they were happy with the system or not.

We anticipate that, over time, the daily ratio of happy vs. sad face presses will increase as the system learns from users how to better interact. We envision a plot that shows this ratio of happiness v/s number of interactions (i.e., collected data) where we should see a point where increasing the interactions no longer improves the system. This is the point that will tell us that what is the amount of data needed to train the model and make the user like it.

CHAPTER 7

CONCLUSION

7.0.1 Dialogue Management

The dialogue manager (DM) plays a crucial role in the SDS [6]: as well as determining how to act, the DM is called upon to decide when to act, effectively giving the DM the control over timing of actions rather than relying on ASR end pointing–further separating our SDS from other systems. The DM policy is based on a confidence score derived from the NLU (in this case we used the distribution's argmax value) using thresholds for the actions set by hand (i.e. trial and error). At each word and resulting distribution from NLU, the DM needs to choose one of the following:

- *wait* - wait for more information (i.e. for the next word.)
- *select* - as the NLU is confident enough, fill the slot with the argmax from NLU.
- *request* - signal a (yes/no) CR on the current slot and the proposed filler.
- *confirm* - act on the confirmation of the user; in effect. *select* the proposed slot value.

This system uses OpenDial here as an IU-model to perform the task of the DM and gives us the opportunity to adjust the values through reinforcement learning. The DM processes makes a decision for each slot, with the assumption that only one slot out of all that are processed will result in a non-*wait* action (though this is not enforced).

The system in [7] improves upon previous work by leveraging the GUI to learn as it interacts. We accomplished this by collecting the words of a completed utterance and corresponding filled slots then informing the NLU that the utterance led to the filled slots-effectively providing an additional positive training example for the NLU. The NLU can then improve its probabilistic mapping between words and slot values; i.e., through updating the sub-model P(U—R) by retraining the classifier with new information. This is a useful addition because the system designer could not possibly know beforehand all the possible utterances and corresponding intents for all users; this effectively allows the system to begin from scratch with little or no training data. In the system we propose here, improving NLU and the DM will allow us to train the model from the data annotated by the user and also help in adjusting the thresholds on which the DM acts, which in combination with grounding will add to the long-term memory component.

7.0.2 Where are the style files?

The file `bsu-ms.cls` contains the formatting directives for the *bsu-ms* style. It is based on the standard LaTeX `report` style with 12 point font option.

1. Simply copy `bsu-ms.cls` to the directory containing your LaTeX document. That way, LaTeX will find it, because it looks in current directory by default.

2. The current style file may be installed in some directory available system wide. (Ask your systems administrator if this is the case). You will have to include that directory in the path LaTeX uses to search for input files. Under Linux, this is controled by the `TEXINPUTS` environment variable, which can be set in the `.bashrc` file in your home directory. For example

```
TEXINPUTS=.:/usr/local/texinputs/:/usr/share/texmf//
export TEXINPUTS
```

adds `/usr/local/texinputs`, a possible location for `bsu-ms.cls`, although it will not take effect until you `source` the `.bashrc` file, or log in again.

3. Install the style files in a directory under your accounts and set the `TEXINPUTS` variable accordingly.

The first or third way is recommended, becaue they involve making a local copy of the style file. This assures your document format will not be affected affected by subsequent updates to `bsu-ms.cls` (but gives you the option to copy the updated file if you want).

CHAPTER 8

THE GREATNESS OF FOO

important line

In short, we use them because they make our lives easier.

as close to the experience of one human interacting with another, as possible.

8.1 Previous work

The greatness of Foo The Great derives from her early work as documented in

8.2 What are their names?

format the references.

8.3 The code of Foo

When showing a program fragment, use the **verbatim** environment. However, when you wan to show an algorithm, use either the **tabbing** or **algorithm** enviornment.

Thesis text is normally "double" spaced. It is customary to single-space

8.4 Other mysteries of Foo

Here is an itemized list of all the mysteries of Foo.

```
import java.awt.Font;
public class Clock extends UpdateApplet public void init() setFont(new Font("TimesRoman",Font.BOLD,24)); resize(400,100);
public void paint( java.awt.Graphics g ) g.drawString( new java.util.Date().toString(), 20, 20 );
```

Figure 8.1: An Embedded Java Clock. This figure also serves as an example of the inclusion of literal code.

- Mystery 1.
- Mystery 2.
- Mystery 3.
- Mystery 4.

Here is a simple table.

Table 8.1: The Approximate Time of Parallelizing Each Code

Parallel library/language	WRS Code	OCS Code	ICSAMD Code
PVM	20 hours	2 weeks	1 month
HPF	3 hours	1 1/2 weeks	1 month

CHAPTER 9

THE NOT SO GREAT THINGS ABOUT FOO

9.1 Figures

Check Figure 9.1 for what happens when Foo gets compiled.

Figure 9.1: How to Correct Errors in a Fuzzy Image

9.2 Tables

Table 9.1 shows the formatting and labeling for a table.

Table 9.1: Complexity of Selection and Search in Sorted Matrices

	Sorted $X+Y$	Matrix with sorted rows and sorted columns		Matrix with sorted columns
	$\lvert X\rvert = \lvert Y\rvert = n$	$n \times m$, $m \leq n$	$n \times n$	$n \times m$
$k = \Theta(mn)$ or $\Theta(n^2)$	$\Theta(n)$	$\Theta(m \log(2n/m))$	$\Theta(n)$	$\Theta(m \log n)$

Here is another table

Table 9.2: Comparison of Slow PVM Version and the Fast PVM Version

Parameters			Process Number						
N	M		1	5	10	15	20	25	30
128	100	Slow PVM(secs)	2.11	3.91	5.78	8.26	10.91	14.17	19.47
		Fast PVM(secs)	2.10	1.20	1.56	1.95	2.79	3.22	4.07

Table 9.3: The Speedup of the PVM WRS Code and the HPF WRS Code

Parameters			Process Number						
N	M		1	10	20	30	40	50	60
128	600	PVM(speedup)	1	5.18	7.67	8.24	6.99	5.55	4.49
		HPF(speedup)	1	8.40	12.15	13.98	14.73	13.52	13.21
256	300	PVM(speedup)	1	6.70	7.74	6.47	5.19	3.72	2.94
		HPF(speedup)	0.99	7.24	9.65	10.58	10	9.48	8.73
512	150	PVM(speedup)	1	6.75	10.64	12.14	13.35	13.87	13.98
		HPF(speedup)	0.98	6.72	9.88	11.55	12.86	13.38	13.83
1024	75	PVM(speedup)	1	2.13	2.30	2.36	2.38	2.39	2.40
		HPF(speedup)	0.95	1.94	2.06	2.10	2.13	2.13	2.14

CHAPTER 10

CONCLUSIONS

10.1 What have we done so far?

10.2 Future directions

The coming revolution in foo-lets offers many opportunities for further research.

REFERENCES

[1] Kent Bach. Meaning , Speech Acts, and Communication. *Basic Topics in the Philosophy of Language*, pages 1–23, 1994.

[2] Joyce Y Chai, Lanbo She, Rui Fang, Spencer Ottarson, Cody Littley, Changsong Liu, and Kenneth Hanson. Collaborative effort towards common ground in situated human-robot dialogue. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pages 33–40, Bielefeld, Germany, 2014.

[3] Herbert H Clark. *Using Language*. Cambridge University Press.

[4] Herbert H. Clark and Susan E. Brennan. Grounding in communication.

[5] Julian Hough and David Schlangen. Investigating Fluidity for Human-Robot Interaction with Real-time , Real-world Grounding Strategies. (September):288–298, 2016.

[6] Casey Kennington and David Schlangen. Supporting Spoken Assistant Systems with a Graphical User Interface that Signals Incremental Understanding and Prediction State. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 242–251, Los Angeles, sep 2016. Association for Computational Linguistics.

[7] Casey Kennington and Aprajita Shukla. A Graphical Digital Personal Assistant that Grounds and Learns Autonomously. 2017.

[8] Kenny R. Coventy, Thora Tenbrink and John Bateman. Definite reference and mutual knowledge. *Elements of discourse understanding*, 2009.

[9] Dan Kondratyuk and Casey Kennington. Towards a Dialogue System with Long-term, Episodic Memory. In *Proceedings of SEMDial*, Saarbrücken, Germany, 2017.

[10] David Schlangen and Gabriel Skantze. A General, Abstract Model of Incremental Dialogue Processing. In *Dialogue & Discourse*, volume 2, pages 83–111, 2011.

APPENDIX A

TIMING MEASUREMENTS

Here is Appendix A. See Appendix B for the experimental setup.

APPENDIX B

EXPERIMENTAL SETUP

Here is Appendix B.