Semi-situated Learning of Verbal and Nonverbal Content for Repeated Human-Robot Interaction

Iolanda Leite, André Pereira, Allison Funkhouser, Boyang Li, Jill Fain Lehman
Disney Research, Pittsburgh, USA
iolanda.leite,andre.pereira, allison.funkhouser, albert.li,
jill.lehman@disneyresearch.com

ABSTRACT

Content authoring of verbal and nonverbal behavior is a limiting factor when developing agents for repeated social interactions with the same user. We present PIP, an agent that crowdsources its own multimodal language behavior using a method we call *semi-situated learning*. PIP renders segments of its goal graph into brief stories that describe future situations, sends the stories to crowd workers who author and edit a single line of character dialog and its manner of expression, integrates the results into its goal state representation, and then uses the authored lines at similar moments in conversation. We present an initial case study in which the language needed to host a trivia game interaction is learned pre-deployment and tested in an autonomous system with 200 users "in the wild." The interaction data suggests that the method generates both meaningful content and variety of expression.

CCS Concepts

•Human-centered computing → Collaborative and social computing systems and tools; Collaborative and social computing; **Collaborative and social computing systems and tools; •Information systems** → *Crowdsourcing;*

Keywords

Long-term human-robot interaction; crowdsourcing; content authoring; multimodal behavior generation.

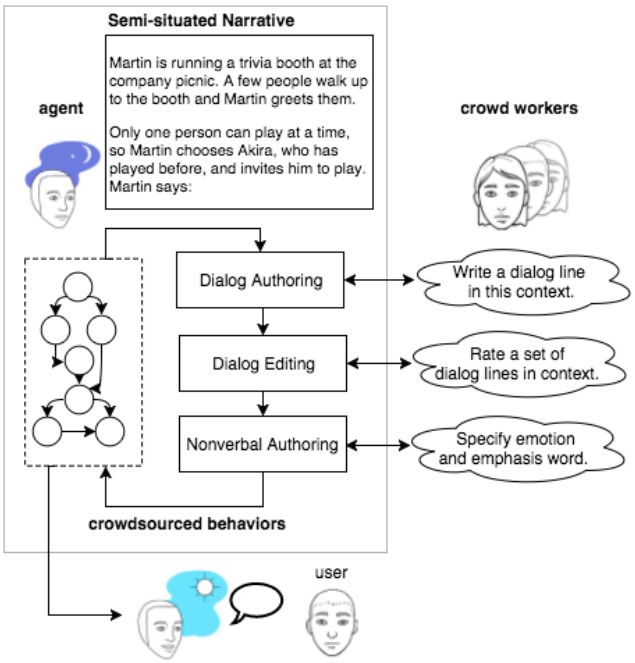


Figure 1: Overview of our semi-situated learning pipeline.

1. INTRODUCTION

One of the main characteristics of human conversation is variety of expression. Even when we interact with people to do the same activity over and over, the language we use is situated in the moment and variable in ways that are not necessary to accomplishing the task. We only need the word *hello* to effectively greet someone, for example, but we also use *hi* in all the same contexts, and *good morning/night*, *hey*, *dude*, or a simple head nod in situations where they are appropriate but *hello* would serve as well. To interact naturally with users over long periods of time, autonomous characters (social agents) should have this variability of expression, too.

A long-standing barrier to extended human-robot interaction is the on-going need to author content – what the robot can do, say, and understand [12]. Insisting on variety of expression would seem to raise that barrier higher. But while we acknowledge that, as system builders, we must author what the character does, we argue that authoring what it says and understands, with all the requisite variety of expression, can be left in no small part to the character itself. To support our position, we present a persistent interactive personality (PIP) that acquires verbal and non-verbal dialog behaviors using *semi-situated learning*, a robot-human pipeline under autonomous control. In essence, PIP renders segments of its goal graph into brief stories that describe situations in PIP's terms, then sends the stories to crowd workers who author and edit a single line of character dialog and its manner of expression given that context. In the final step, PIP integrates the result of the process back into its goal graph, and uses the elicited language behavior when it is in conversation with a user in a similar state.

The approach has three main advantages over content-authoring by the system builder's hand. First, it produces variation by eliciting language samples from an array of individuals, each of whom

is likely to differ in his/her natural forms of expression from the others and from the system builder. Second, the language from those individuals is nevertheless likely to be meaningful *in situ* because it is generated by writers who bring human understanding and experience to a story that is explicitly focused on task-relevant features. Lastly, it provides a uniform framework for two modes of agent building: the acquisition of a static set of multimodal language behaviors pre-deployment, and the incremental acquisition of such behaviors over time as repeated interaction with an individual becomes stale or PIP's domain of discourse is extended. The former is useful for rapid prototyping, while the latter is important for long-term interaction.

After a short discussion of related work, we describe the method PIP uses to acquire language behaviors in detail. We then present an initial case study in which the language needed for the interaction is learned as a static dialog capability pre-deployment, and tested in an autonomous system "in the wild." With that experience as a concrete basis, we conclude with a discussion of how the framework allows for a continuum of "situatedness" and our plans to explore that continuum in the future.

2. RELATED WORK

Research on social robots and agents supporting repeated interactions with the same user has increased in the past years [10, 2, 11, 20, 6]. Typically, engineers, animators or knowledge experts have been responsible for authoring the agent's verbal and nonverbal behavior, a process that does not scale well when aiming for variability across interactions that span weeks and months. Crowdsourcing has been proven to be a successful approach for acquiring large amounts of non-expert language data in a variety of tasks [21], from labeling objects on an image [19] to generating alternate paths for a narrative [13]. Our approach builds on and extends both these areas of interest.

One of the pioneer systems exploring the use of crowdsourcing for interactive characters is the Restaurant Game [16], a virtual world where human players interacted in pairs, with one in the role of waiter and the other as a customer avatar. Using the human player logs, the authors built a plan network – a statistical model that encodes contextual patterns of behavior and language – and used it to generate an artificial agent that mimics human behavior in a restaurant setting [17]. The same approach was followed by Breazeal et al. [3] to drive the behavior of a robot in a cooperative problem-solving activity. The autonomous robot behavior, generated from data collected from humans performing the task, was compared to the behavior produced by a human operating the robot in a user study. The authors found that participants rated both robot conditions favorably. There are two important features that distinguish these works from ours. First, both systems require at least two players interacting simultaneously and completing an entire task, while in our case each crowd worker is assigned to a small unit of a task and can perform asynchronously with respect to the other workers. Second, while the focus here is on learning sequences of actions that an agent can take, our work assumes predefined action sequences and focuses on the process of obtaining variability of expressive behavior in a given situation.

The use of crowdsourcing methods has also been explored in spoken dialog systems [7, 22]. One such example is Chorus [9], a conversational assistant that provides online help based on the combined efforts of multiple crowd workers interacting with a user in real-time. Chorus is presented as a collaborative reasoning system because it allows crowd workers to add new responses to user questions, select the top responses (from among those provided by other workers), or remove responses that do not fit the flow of the conversation. More relevant to our work, Mitchell *et al.* created a pipeline for generating paraphrases from an existing corpus of dialog using crowdsourcing [15]. An evaluation of the paraphrases showed that crowd workers were able to provide diverse alternatives, still suited to the dialog context. We extend this idea with an architecture that allows the agent to crowdsource its entire dialog content from scratch.

While many authors have used crowdsourcing methods for verbal dialog-related tasks, less attention has been given to ways of authoring an agent's nonverbal behavior. An exception is Rossen and Lok, who investigated the authoring of both verbal and nonverbal content using crowd-based methods [18]. They developed an authoring tool that facilitates collaborative development of virtual humans by two groups of end-users: domain experts (educators) and domain novices (students). This is one of the few examples where crowdsourced data supported the integration of animations and emotional behavior in a dialogue system. However, experts are still the main content authors and editors, while our system is intended for non-expert crowd workers who contribute to a larger, autonomously-controlled agent functionality.

3. AN ARCHITECTURE FOR SEMI-SITUATED LEARNING

PIP generates its own language capability off-line by systematically exploring goal-state descriptions of the situations it might find itself in, recasting those descriptions into a story form that is easy for people to understand, and crowdsourcing the production of a meaningful dialog line at the end of each narrative. Despite its context-dependence, the learning is only semi-situated because the character's state at the moment it speaks the line is likely to include more information than the narrative expressed. Nevertheless, by using goal-specific features to generate the story and human crowd workers to author the dialog, the character's resulting language behavior tends to be both meaningful in the moment and globally coherent.

In this section we describe PIP's process in detail, left-to-right and top-to-bottom through Figure 1. We begin with a description of the internal representation it needs to generate narratives, then progress through the three autonomously-controlled pipeline components: dialog authoring, dialog editing, and nonverbal behavior authoring.

3.1 Generating Narratives for the Pipeline

Crowd-sourced dialog needs an underlying representation of its internal structure to be successfully associated with valid agent goals and employed in the proper context [17]. As a conversational agent, PIP has a *goal state graph* that both outlines the high-level steps of an interaction between itself and a user and specifies variables whose values drive transitions between those steps. To elicit language that will be useful *in situ* we add meta-information to the nodes in the goal state graph that makes explicit the set of *contextual variables* that should be represented in the crowd workers' narrative as well as the text strings to be used in composing it. The meta-information comes in two forms:

- *Synopsis*: unparameterized text that is used to summarize a conversational goal and appended to the narrative as-is. The root node of the graph contains a special synopsis, the *exposition*, which gives the initial description of the setting where the story takes place and relevant information about the people involved. In the example narrative at the top of Figure 1, the exposition corresponds to the first sentence, "Martin is running a trivia booth at the company picnic."

In non-root graph nodes, the synopsis is a generic encapsulation of goal information that helps to move the story to the point where dialog is needed. In Figure 1, the sentence "A few people walk up to the booth and Martin greets them," is a synopsis of the *greet multiple players* goal (detailed in Figure 3). Note that the synopsis does not specify exactly what Martin said in greeting or what state variables PIP would have used to choose that dialog line; it only establishes that the greeting step has been accomplished.

- *Narrative template*: a node-specific data structure that is used to generate the portion of the narrative for which a dialog line is to be elicited. The template is instantiated by binding each of its contextual variables to a specific value and adding the associated text to the story. The second paragraph in Figure 1 shows one instantiation of the narrative template for the goal *single out player*. The sentence "Only one person can play at a time, so Martin chooses Akira, who has played before, and invites him to play, saying..." describes the goal in story terms and includes the phrase "who has played before," which is associated with the values **one** and **more** for the variable **num_interactions**. **Num_interactions** can also take the value **0**, which would instantiate the string "who has never played before" in the template instead. Note that each goal in the graph can have multiple templates and each template can have multiple contextual variables.

In this example, taken from the version of PIP described below, we chose to collapse two possible values for **num_interactions** into a single template option. This is one example of why the learning that occurs is semi-situated – when PIP speaks a line elicited at this node, the state will not only record whether the user has played once or more than once, that nominative value will have been derived from data that indicates how many times the user has played. The content author, on the other hand, knows only that previous play has occurred. Although consistency is maintained, to the extent that the distinction might have made a difference in what the author wrote, potential, meaningful variability is lost.

More broadly, the degree of "situatedness" in the narrative will always need to be traded off against the combinatorics of acquisition as a system-building decision. To use the meta-information we give it to acquire dialog, PIP does a repeated, top-down graph traversal from the root node, for paths with length greater than or equal to one. Each partial path generates either a single narrative (if the template of the final node has no contextual variables) or multiple narratives (one for each possible combination of contextual variables and their values). In cases where the graph's combinatorics are small enough and it is desirable for all the character's dialog to be determined in advance, the traversal can be exhaustive. This is the approach we took in the case study, giving PIP a relatively small number of templates per node, each of which was predicated on a relatively small number of state variables and values. As a contribution to the eventual dialog capability, this amount of content authoring was minimal.

Although our approach is intended to expand the dialog possibilities offline, it is not necessary that it do so all at once. Because each path produces a narrative that begins with the exposition and continues synopsis-by-synopsis to a particular point in the interaction, PIP can also acquire its dialog functionality incrementally by expanding depth-first over a subset of the templates at each node. At any given point in time, the resulting system would be functional but have less variation of expression available to it. In theory, expanding incrementally could also mean expanding opportunistically, a point we return to when discussing future work.

3.2 Crowdsourcing Dialog

Every narrative that is generated by the above procedure is sent into the crowd worker pipeline shown at the right in Figure 1. Currently, PIP uses the API for Amazon Mechanical Turk (AMT) to automatically manage the requests and results of the Human Intelligence Tasks (HITs) that correspond to dialog authoring, dialog editing, and nonverbal behavior authoring. To elicit dialog that is as natural sounding as possible, narratives are written with PIP identified as "Martin" and not as a robot. In addition, the templates randomly assign a name to the other character from a set that has been chosen to counterbalance for possible cultural and gender effects in the language. When parsing the dialog authoring results, PIP replaces any repetition of the story characters' names with **agent** and **user** tags. Given this general structure, we turn to the tasks themselves.

Dialog Authoring: The first task, *dialog authoring*, consists of reading the narrative and writing a single line of dialog at the prompt at the end (e.g.,"Martin says:"). To increase variability of expression, tasks are constrained such that a worker can provide only one line of dialog for a given narrative. PIP has one system parameter that determines how many different workers should receive the authoring task, and another parameter for how many different lines for the same story should be bundled together for the editing phase.

Dialog Editing: In the second stage of the pipeline, editors judge the quality of authored dialog lines. In particular, a new group of workers is tasked to read the same narrative the authors saw, and for each dialog line either flag it as "nonsensical" or rate it on a scale from 1 ("makes sense, but I wouldn't say this") to 5 ("I would totally say this"). As a quality check on the editors' performance, a truly nonsensical utterance is inserted into the authored lines in every set. If a worker fails to mark that nonsensical utterance as such, his/her judgments are excluded and another worker is recruited until the desired number of editors is met.

After obtaining the required number of valid judgments, PIP processes the results before generating the final set of HITs. The system only passes along those dialog lines that: (1) were considered as nonsensical by no more than one editor, and (2) have an average score greater than 2. In pilot tests of the pipeline, we found that three judges were enough to get both reasonable agreement on the utterances to discard and reasonable quality on the utterances to keep.

Nonverbal Behavior Generation: The final step of the pipeline elicits information that can be used by PIP to program its nonverbal behavior when speaking editor-approved dialog. In particular, we ask multiple crowd workers to assign both a point of emphasis and an emotion to each line, given the same narrative context in which the line was authored.

To elicit a point of emphasis, nonverbal authors are asked to read the dialog line out loud to themselves and mark the word in the sentence that received the most verbal emphasis. Although PIP does not currently program its own prosody, it can use the information resulting from these HITs to self-program subtler gestural indicators, like blinks and eyebrow movements, that are correlated with verbal emphasis [5].

Nonverbal authors are also asked to select the emotion that would best match the dialog line in context. They do so from a long drop-down list of possibilities – excited, happy, smug, surprised, bored, confused, skeptical, embarrassed, concerned, sad, and neutral (in case none of the other emotions seems suitable) – in order to exploit the variability of expressive behavior our robot affords.

Once the scheduled number of workers completes the task, simple rules are used to map the results to nonverbal behaviors. If the

Figure 2: PIP interacting with a player during the trivia game in the 3-day office deployment.

most commonly chosen emphasis placement receives at least 75% of the selections, an emphasis gesture is added to that location and performed at the appropriate time during speech synthesis. If there is no dominant word but the two most commonly chosen locations are adjacent and form a noun/adjective or verb/adverb pair, and if the two words together receive at least 75% of the participant selection, an emphasis gesture is added to span the phrase. Otherwise, no emphasis is added to the dialog line.

Similarly, if the most commonly chosen emotion receives at least 70% of worker selections, the dialog line is annotated with the emotion and PIP adopts the corresponding expression when the line is uttered. Otherwise, if the two most commonly chosen emotions have the same valence (e.g., concerned and sad), and if the two emotions together receive at least 70% of the selections, the expression with the highest percentage is added. If neither of these two conditions is satisfied, the expression remains neutral.

In both the pilot study that established the validity of the nonverbal tasks [4] and the case study presented in the next section, we found that a set of 10 workers was sufficient for the emphasis and emotional expressions to converge almost 100% of the time using the specified thresholds of 75% and 70%, respectively. Varying the thresholds might lead to a different optimal number of workers for this task.

The end product of the pipeline is a set of annotated, human-authored dialog lines that are associated with the goal that generated them and indexed by the values of a subset of the state variables for that goal. The annotations include both the nonverbal directions for expression and the average score from the editing phase. Under the assumption that higher scores are associated with more natural sounding utterances, PIP uses that information as part of its policy in selecting a line from the alternatives available when in conversation. In the next section we explore the results of semi-situated learning in practice.

4. CASE STUDY

To provide a proof-of-concept for the semi-situated approach, we have implemented a repeated-interaction scenario in which most of PIP's behavior was authored by the crowd. In the scenario, PIP plays the role of a trivia game host. The game consists of players listening to a brief audio clip and trying to guess which of a small set of movies the clip corresponds to in order to win a point for their team. The interactions are short, and the competitive aspect of the game was chosen to motivate players to come back and interact with the character multiple times.

4.1 Quizmaster PIP and the Trivia Game

PIP was embodied in a Furhat robot head [1] atop a stationary wooden form (see Figure 2). The Microsoft Kinect V2 and far-field RFID antenna behind PIP enable the character to track and recognize individual users, a critical ability given our goal of repeated interactions and our use of contextual variables that depend on them. The identification subsystem matches a skeleton tracked by the Kinect to a unique RFID tag worn by the user [14], allowing PIP to recognize in a few seconds individuals in a group of up to six people with 95% average accuracy. A near-field RFID antenna is hidden under the area of PIP's stand marked by the green felt tray; it reads the passive RFID tags in the cards used to play the game.

The interaction's flow can be read from the graph in Figure 3. When PIP recognizes one or more players nearby, it invites one of them to play. If the person has never played before, PIP continues by explaining the game and asking her/him to pick up a set of five movie cards from the tray (with repeat players no instructions are given and the prompt to pick up the cards is used only if a long period elapses without activity). The character then randomly selects a movie quote, apprises the player of its difficulty level, and plays the clip.[1] When the user places a card on the tray, PIP provides feedback on the answer. If the player's guess leads to changes in the overall team scores (e.g., a different team has taken the lead), PIP calls attention to the fact before saying goodbye to the current player. If there are other players waiting nearby, PIP invites one of them to go next.

The choice to use RFID-tagged cards rather than voice input was deliberate. The interaction was intended to be deployed in multiple locations, at least two of which were known to be far noisier than current automatic speech recognition can handle accurately without a close-talk microphone. Thus the current case study explores our approach to dialog acquisition only for PIP's side of the conversation; we return to the question of its role in anticipating the user's language in the Conclusion.

4.2 Language Acquisition

Quizmaster PIP's verbal and nonverbal behaviors were predicated on the goal state graph and contextual variables presented in Figure 3. After we defined the synopses and templates for each node, PIP generated dialog authoring HITs through exhaustive graph traversal, with each narrative sent to ten AMT workers. The elicited dialog lines were separated into two sets of five and each set sent with its narrative to three new workers for the editing phase of the pipeline. Lines not eliminated during editing were then passed, with their narrative, to ten workers who authored the nonverbal emphasis and emotion annotations. Recruited AMT workers were at least 18 years old and were registered in the United States in order to increase the likelihood of acquiring language from native English speakers. Workers were compensated 20¢ per HIT. The aver-

[1]The 170 quotes available in the game were categorized as easy or hard based on an online survey conducted to sort the quotes by difficulty.

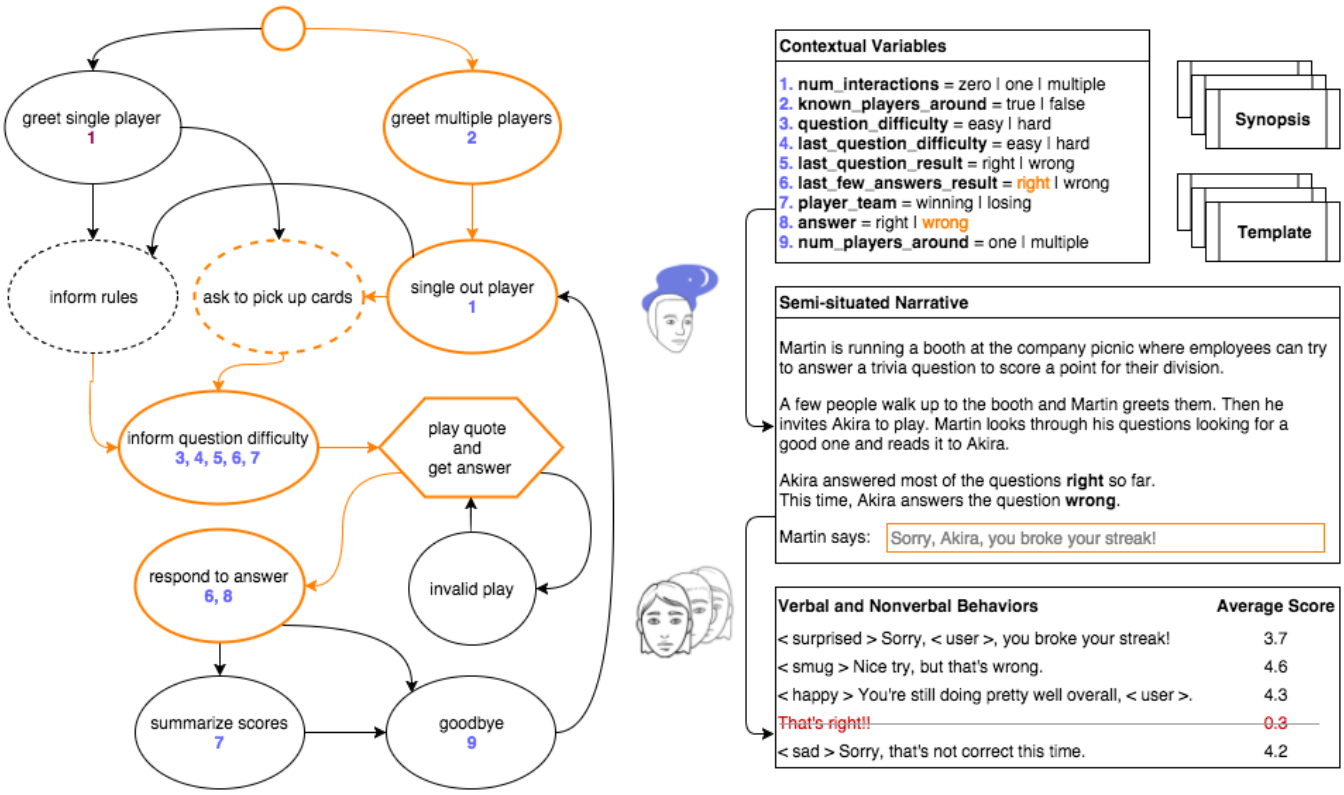


Figure 3: PIP's goal state graph for the trivia game (left) including the number for the contextual variables used to generate narratives in the table at the top-right. Dashed lines represent conversational goals with language that was not authored by the crowd (some error handling nodes omitted for clarity). The right side contains an example of a semi-situated narrative generated by traversing the colored path through the graph to the node *respond to answer*. Below the narrative, some of the language behaviors produced at the end of the pipeline for that story. The line "That's right" was rejected in the editing phase; all the other lines were added to PIP's behavioral repertoire.

age HIT completion time was 1.5 minutes, irrespective of phase of the pipeline.

As shown by the dashed lines in Figure 3, we authored the dialog lines in the goals related to the rules of the game and the use of the cards to communicate. We did this to make sure that all players received the same, correct information about how to play. The goal state graph also includes two nodes, omitted for clarity, that perform error handling related to the tracking system. Their language was also hand-authored by us.

The right side of Figure 3 continues the explanation of how the pipeline works in a concrete situation. One of the semi-situated narratives that workers read when authoring behaviors for the *respond to answer* node appears in the middle. This particular version is predicated on the case where the player answered most of the previous questions correctly (**last_few_answers_result** = right) but answers the current question incorrectly (**answer** = wrong). The first paragraph of the generated narrative contains the exposition obtained from the root node. The second paragraph aggregates the synopses of the *greet multiple players*, *single out player* and *inform question difficulty* nodes (skipping *inform rules* and *ask to pick up cards* because the player will have played before). Finally, the third paragraph includes an instantiation of the *respond to answer* template, with the values of the contextual variables shown in boldface. The table below the narrative contains a sample of the multimodal behaviors authored by the crowd-worker pipeline. Note that where a dialog author used the story character's name in her/his response, the "<user>" tag is substituted. The fourth line is rejected based on average editor score, but the remaining lines become part of Quizmaster PIP's repertoire, and will be considered for speech synthesis when PIP needs to acknowledge a wrong answer by a player who got the last few questions right. Like this example, most of the crowdsourced utterances in PIP's repertoire were elicited with narratives instantiating history-dependent context variables. Table 1 provides additional examples and gives a sense of both the coherence of the resulting dialogs and how PIP's language changes over repeated interactions.

Exhaustive traversal of the quiz game's graph elicited 680 crowdsourced lines of language behavior. A total of 48 (7%) were eliminated either during editing by crowd workers or post-editing by PIP on the basis of average score. The remaining 632 were combined with 84 sentences we created for the instruction goals and related error-handling behaviors. The resulting set of 716 lines of dialog constituted PIP's language capability for the three deployments. Of course, PIP's language behavior in conversation depends on both its stored dialog lines and its policy for selecting among the possibilities. Because we expected that PIP would be approached by small groups in all of the venues, we implemented a policy that cycled through the ranked lines available in each goal state to make it unlikely that observers would hear the same dialog when their turn came to play.

Table 1: An example of how PIP's language changes as a function of the history of interactions with a user. The table shows the verbal and nonverbal behaviors employed by PIP in each goal state while interacting with a user the first, second, and fifth times.

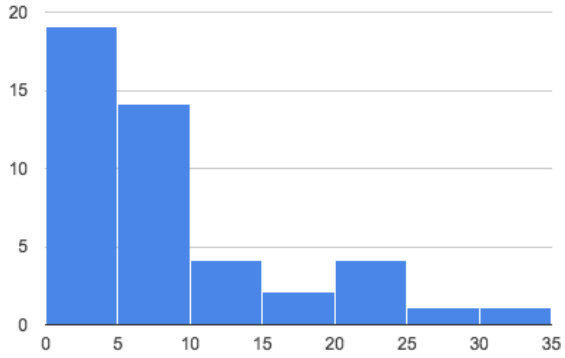
	1st interaction	
goal state	**context variables**	**behavior**
greet single player	num_interactions = zero	< *neutral* > Hello, do you want to play a round of trivia?
inform rules *		I will play a dialogue line from one of the movies in these cards and your goal is to guess which movie. When you know the answer, place the card on the tray.
inform question difficulty	difficulty = easy last_question_difficulty = null last_question_result = null player_team = winning	< *happy* > Let's start with an easy one!
respond to question	answer = right	< *excited* > You got it, < *user* >!
summarize scores	last_few_answers = null player_team = winning	< *player_team* > blowing all the competition out of the water and is ahead!
goodbye	num_players = one	< *happy* > Good job, < *user* >.
	2nd interaction	
goal state	**context variables**	**behavior**
greet single player	num_interactions = one	< *surprise* > Back again, < *user* > ?
ask to pick up cards*		< *look_down* > Please pick up all the cards.
inform question difficulty	difficulty = hard last_question_diffulty = easy last_question_result = win player_team = winning	< *excited* > You're gonna have to try hard for this one!
respond to question	answer = wrong last_few_answers = right	< *happy* > Well, you can't win them all I suppose!
summarize scores	player_team = losing	< *player_team* > needs to step up!
goodbye	num_players = one	< *smug* > Better luck next time.
	5th interaction	
goal state	**context variables**	**behavior**
greet multiple players	known_players = true	< *exited* > Hey guys, ready to test your trivia knowledge?
single out	num_interactions = multiple	< *happy* > < *user* >, would you like to play again?
ask to pick up cards *		When you're ready, pick up the cards.
inform question difficulty	difficulty = easy last_question_difficulty = hard last_question_result = lost player_team = losing	< *excited* > You should be able to help your team with this one.
respond to question	answer = right last_few_answers = right	< *excited* > Right again, < *user* > !
summarize scores	player_team = winning	< *player_team* > has just taken the lead!
goodbye	num_players = multiple	< *excited* > Let's keep this party rolling, who wants to go next?

4.3 System Deployment

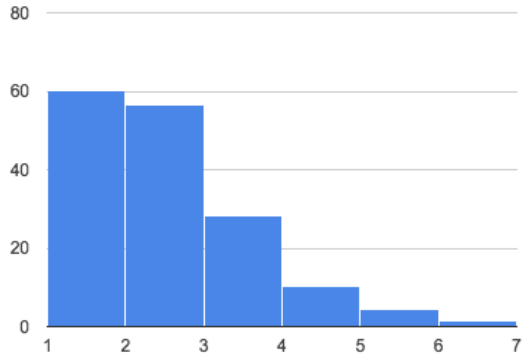
PIP hosted the quiz game in three separate locations: an office contest and two public events. In the office deployment, the game was available for repeated interactions over three days. The public events consisted of half-day research showcases in different places with non-overlapping attendees. No audio or video was recorded and no personally identifying information collected; anyone who wanted to play simply picked an RFID tag to use for the duration of the deployment period. Each tag was associated with a fictional name and a color that represented one of the teams so that PIP could address each player personally when a dialog line required it. Most players interacted with PIP alone or in groups of two.

We calculated statistics separately for each type of environment based on the interaction logs. Office data was generated by 41 unique players, with an average of 8.4 interactions per player ($SD = 8.2$), and a 36.8 second mean length of interaction ($SD = 10.3$). About 90% of office players returned to play again; the distribution of number of plays can be seen in Figure 4a. The public deployments were, not surprisingly, quite different, with 160 unique players across the two locations averaging 2.1 rounds each ($SD = 1.1$), and 44.5 second mean length of play ($SD = 21.3$). With many other projects to see and a limited time in which to see them, only 62.5% of showcase attendees chose to play more than once. The distribution of return rate appears in Figure 4b.

Although our purpose in creating a contest was to encourage repeat play, we were far more successful in the office environment than we had expected. As a result, it was inevitable that some users in that deployment would experience repeated utterances, given that there were at most ten alternatives in a goal state. Figure 5 shows how users experienced a decay in the variability of expression as the number of times they played increased. Players at the public events heard almost entirely unique language each time they played, even though there was often significant delay between their games. Moreover, despite the fact that the more competitive players in the office environment did hear individual lines repeated, no player ever heard exactly the same dialog twice.



(a) Office.



(b) Public events.

Figure 4: Number of players (Y axis) as a function of how many times they interacted with Quizmaster PIP (X axis) in the different deployments. Note that the units on the axes are not the same across locations.

4.4 Observations and Discussion

We could have used the time between the acquisition of PIP's dialog behavior and its use in public to sort through the crowdsourced results, eliminate utterances we did not like, and elicit additional ones.[2] Instead, we made the conscious choice to send PIP into the wild with only what semi-situated learning provided. Nevertheless, a *post hoc* analysis of the pipeline's results showed that, for the quiz game's goal state graph, the process was both more and less critical than we would like it to be.

On inspection we felt that only 17 of the 48 utterances rejected by the editing phase were unusable. While the overall yield of 93% is still excellent (and well beyond the level typically associated with Mechanical Turk [8]), the questions remain of whether and how the parameter settings could be improved and, more importantly, whether and how they might be automatically derived, given a new interaction task and goal state graph.

A small percent of over-constraint has no impact on the user's experience, but even a small percent of under-constraint can. We use narratives to take advantage of the normative social knowledge that our authors have – knowledge that they use in deciding which features of the story to articulate when they write a line of dialog. In some cases, however, such knowledge introduces language that is in conflict with PIP's goal state. We found three examples, stemming from two distinct sources. The first type is exemplified

[2]Certainly, had we known that people would play 20 or 30 times we would have increased the number of workers requested and/or the number of templates we wrote.

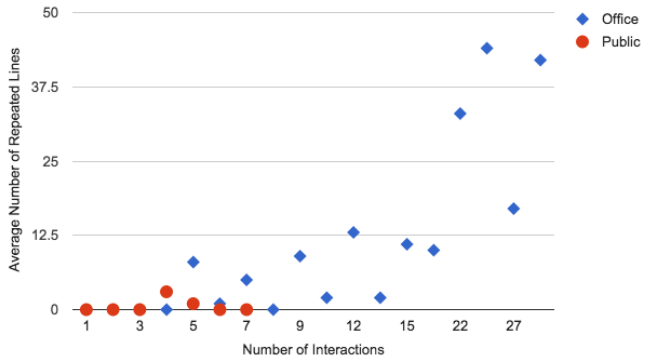


Figure 5: The average number of crowd-authored behaviors that users heard repeated as a function of the number of trivia games they played (note that the X axis is non-linear).

by the crowdsourced line, "Attention everyone, that is the man to beat." In this case, the author did exactly what we wanted – picking up on a narrative detail (the gender of the non-Martin character) – and the editors agreed with the detail when validating the line in the same story context. Ultimately the line is problematic, however, because we chose not to encode gender as a context variable in the templates to avoid having to detect or collect that information about the players. Without the contextual variable, the line is inadvertently generalized across all players, and used regardless of the interacting player's actual gender. The second source of missituatedness is exemplified by the lines, "Sorry, you can only try once" and "I'm sorry, you can only play 2 times max!" In this case, the author added information that was relevant to a contextual variable (**num_interactions**), normative enough to be accepted by the editors, but simply not true. Although altogether these kinds of utterances constitute less than half a percent of PIP's repertoire, additional oversight, either prior to use or by the player, would be required to remove them.

5. CONCLUSIONS AND FUTURE WORK

We have presented a narrative-based approach to authoring dialog behavior and demonstrated through an initial case study that it generates both meaningful content and variety of expression. The image of PIP we are working toward is one where a highly task-specific, goal-directed interaction is just one capability among many and, more importantly, interleaved with more natural social discourse. While we consider these preliminary results promising we acknowledge that our method could be extended in at least three important ways.

First, the choice to avoid speech input was intentional but temporary. Indeed, one of the potential advantages to semi-situated learning is that it can be used to elicit dialog for the user as well as the robot. We intend to explore whether user dialog authored via the same pipeline can form the basis of a lexical semantic model to augment automatic speech recognition. We know from The Restaurant Game that this is likely to be useful in a well-specified, goal-constrained task; we are interested in whether it can be accomplished autonomously and incrementally as well as whether it will work for a more diffuse conversational context like social chit chat.

Second, our experience with Quizmaster PIP makes it clear that some dialog is likely to make it through the pipeline that reflects distinctions that are missing from the context variables in the template or the state. The result will be speech that is at odds with

the situation in which it is uttered. Although this occurred very rarely in the case study, the particular instances we encountered are part of a larger theoretical problem. Just because a contextual variable is articulated in the narrative does not guarantee that the author will generate a dialog line that depends on it. Yet the process described in this paper will assume that the author did so, storing it away as a function of the instantiated variables that elicited it. In general, then, the indexical terms associated with a dialog line may be broader or narrower than the line's actual applicability in the interaction space. This suggests that PIP could use its situated experiences – which always occur with its entire state space instantiated – to modify the indices over time. We are exploring augmenting the current pipeline with a combination of active hypothesis-testing and user-controlled negative reinforcement. Hypothesis testing would broaden PIP's selection policy, allowing it to choose a line in a state that does not match its indices (can it use the line more generally?) as well as track the state variables where it has and has not received negative feedback (should it use the line more restrictively?). The same hypothesis-testing extension could also be used off-line with the second phase of the pipeline, embedding a dialog line authored under one contextual value in a narrative that varies that value and letting editors judge the line's goodness.

Finally, the length and range of interactions we want PIP to be able to engage in is much broader than the one presented in our case study. While the trivia game deployments discussed here used exhaustive graph traversal over a small set of templates and state features, in a sufficiently large state space and/or over longer periods of time PIP should be able to expand its graph more opportunistically. This could be done incrementally (by sending new HITs out every night) or as a function of experience (biasing expansion to the areas of the graph where most of the interaction seems to be taking place). An optimization function that allows PIP to use its resources (money, moments of actual interaction) to best effect is a clear direction for future work.

6. REFERENCES

[1] S. Al Moubayed, J. Beskow, G. Skantze, and B. Granström. Furhat: a back-projected human-like robot head for multiparty human-machine interaction. In *Cognitive behavioural systems*, pages 114–130. Springer, 2012.

[2] T. Bickmore, D. Schulman, and L. Yin. Maintaining engagement in long-term interventions with relational agents. *Applied Artificial Intelligence*, 24(6):648–666, 2010.

[3] C. Breazeal, N. DePalma, J. Orkin, S. Chernova, and M. Jung. Crowdsourcing Human-Robot interaction: New methods and system evaluation in a public environment. *Journal of Human-Robot Interaction*, 2(1):82–111, 2013.

[4] A. Funkhouser. Annotation of utterances for conversational nonverbal behaviors. Master's thesis, Robotics Institute, CMU, May 2016.

[5] H. P. Graf, E. Cosatto, V. Strom, and F. J. Huang. Visual prosody: Facial movements accompanying speech. In *Proc. of the 5th IEEE International Conference on Automatic Face and Gesture Recognition*, pages 396–401. IEEE, 2002.

[6] A. D. Jacq, S. Lemaignan, F. Garcia, P. Dillenbourg, and A. Paiva. Building successful long child-robot interactions in a learning context. In *Proc. of the 11th ACM/IEEE International Conference on Human Robot Interaction*, pages 239–246. IEEE Press, 2016.

[7] F. Jurcıcek, S. Keizer, M. Gašic, F. Mairesse, B. Thomson, K. Yu, and S. Young. Real user evaluation of spoken dialogue systems using amazon mechanical turk. In *Proc. of INTERSPEECH*, volume 11, 2011.

[8] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proc. of the SIGCHI conf. on human factors in computing systems*, pages 453–456. ACM, 2008.

[9] W. S. Lasecki, R. Wesley, J. Nichols, A. Kulkarni, J. F. Allen, and J. P. Bigham. Chorus: a crowd-powered conversational assistant. In *Proc. of the 26th annual ACM symposium on User interface software and technology*, pages 151–162. ACM, 2013.

[10] M. K. Lee, J. Forlizzi, P. E. Rybski, F. Crabbe, W. Chung, J. Finkle, E. Glaser, and S. Kiesler. The snackbot: documenting the design of a robot for long-term human-robot interaction. In *Proc. of the 4th ACM/IEEE International Conference on Human-Robot Interaction*, pages 7–14. IEEE, 2009.

[11] I. Leite, G. Castellano, A. Pereira, C. Martinho, and A. Paiva. Empathic robots for long-term interaction. *International Journal of Social Robotics*, 6(3):329–341, 2014.

[12] I. Leite, C. Martinho, and A. Paiva. Social robots for long-term interaction: a survey. *International Journal of Social Robotics*, 5(2):291–308, 2013.

[13] B. Li, S. Lee-Urban, G. Johnston, and M. Riedl. Story generation with crowdsourced plot graphs. In *Proc. of the AAAI Conference on Artificial Intelligence*, 2013.

[14] H. Li, P. Zhang, S. Al Moubayed, S. N. Patel, and A. P. Sample. Id-match: A hybrid computer vision and rfid system for recognizing individuals in groups. In *Proc. of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 7–7, New York, NY, 2016. ACM.

[15] M. Mitchell, W. Redmond, D. Bohus, and E. Kamar. Crowdsourcing language generation templates for dialogue systems. In *Proc. of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2014.

[16] J. Orkin and D. Roy. The restaurant game: Learning social behavior and language from thousands of players online. *Journal of Game Development*, 3(1):39–60, 2007.

[17] J. Orkin and D. Roy. Automatic learning and generation of social behavior from collective human gameplay. In *Proc. of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 385–392. IFAAMAS, 2009.

[18] B. Rossen and B. Lok. A crowdsourcing method to develop virtual human conversational agents. *International Journal of Human-Computer Studies*, 70(4):301–319, 2012.

[19] O. Russakovsky, L.-J. Li, and L. Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2121–2131, 2015.

[20] E. Short, K. Swift-Spong, J. Greczek, A. Ramachandran, A. Litoiu, E. C. Grigore, D. Feil-Seifer, S. Shuster, J. J. Lee, S. Huang, et al. How to train your dragonbot: Socially assistive robots for teaching children about nutrition through play. In *Proc. of the 23rd IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 924–929. IEEE, 2014.

[21] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 254–263, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[22] Z. Yu, Z. Xu, A. Black, and A. Rudnicky. Chatbot evaluation and database expansion via crowdsourcing. In *Proc. of the chatbot workshop of LREC*, 2016.