Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Learning to Interpret Natural Language Commands through Human-Robot Dialog

Jesse Thomason and **Shiqi Zhang** and **Raymond Mooney** and **Peter Stone**
Department of Computer Science, University of Texas at Austin
Austin, TX 78712, USA
{jesse,szhang,mooney,pstone}@cs.utexas.edu

Abstract

Intelligent robots frequently need to understand requests from naive users through natural language. Previous approaches either cannot account for language variation, e.g., keyword search, or require gathering large annotated corpora, which can be expensive and cannot adapt to new variation. We introduce a dialog agent for mobile robots that understands human instructions through semantic parsing, actively resolves ambiguities using a dialog manager, and incrementally learns from human-robot conversations by inducing training data from user paraphrases. Our dialog agent is implemented and tested both on a web interface with hundreds of users via Mechanical Turk and on a mobile robot over several days, tasked with understanding navigation and delivery requests through natural language in an office environment. In both contexts, We observe significant improvements in user satisfaction after learning from conversations.

1 Introduction

It is important for intelligent robots to be able to efficiently and accurately understand instructions from naive users using natural language. Many existing natural language instruction approaches either use simple language understanding (e.g., keyword search), or large corpora of hand-annotated training data to pair language with robot actions or action language. The former cannot account for naive user language variation. The latter requires gathering annotated corpora, which can be expensive and can only account for variation observed in the training data. This paper presents a dialog agent that communicates with users through natural language while learning semantic meanings from conversations.

Our dialog agent integrates a semantic parser producing logical form representations of user utterances with a dialog manager that maintains a belief-state for the user's goal. The agent starts with a few training examples for the parser and induces more during natural clarification dialogs with ordinary users. When the agent understands a user goal, it pairs the logical form representing that goal with previously misunderstood utterances in the conversation to form new training examples for the semantic parser. This allows the agent to incrementally learn new semantic meanings for previously unseen words. This approach is more robust than keyword search and requires little initial data. Further, it could be deployed in any context where robots are given high-level goals in natural language.

We demonstrate through hundreds of conversations from human users through Mechanical Turk[1] that the agent's learning abilities help it to understand and not frustrate users while converging to goals quickly. However, users interacting with a live robot introduce lexical variations that may be user or task-specific, and do not allow for the contextual control (e.g. linguistic priming, detecting malicious users) afforded by a web interface like Mechanical Turk. We embody the agent in a robot in our office and find that, even from such uncontrolled in-person conversations, it improves understanding and is less frustrating after a brief training period.

To the best of our knowledge, our agent is the first to employ incremental learning of a semantic parser from conversations on a mobile robot. As a result, our robot may have the most robust lexical acquisition capability of any to date.

2 Related Work

Researchers have explored using natural language to inform and instruct robots. Meriçli *et al.* [2014] allow users to specify a task program to be stored and executed on the robot. Like our dialog agent, their system prompts users to correct its (mis)understandings. However, their natural language understanding is done by keyword search and assumes certain words in a particular order. Our dialog agent uses a richer, semantic understanding. Robot world knowledge can also be updated, such as using semantic parsing to extract an action, pre- and post-world conditions for that action, and the entities involved [Cantrell *et al.*, 2012]. The goal of that work is different from ours and its parser is trained on an existing dataset (CReST [Eberhard *et al.*, 2010]), in contrast to our induced training data.

Natural language instruction can dictate a series of actions to a robot. Some approaches pair robot actions with language descriptions, then build models that map language instructions to action sequences [Misra *et al.*, 2014; Tellex *et al.*, 2011]. We are concerned with interpreting high-level instructions rather than action sequences and don't rely as they do

[1] `https://www.mturk.com`

on a well-trained initial parser [Klein and Manning, 2003]. Another approach enables a robot to learn a sequence of actions and the lexical items that refer to them from language instruction and dialog [She *et al.*, 2014]. We focus on acquiring new lexical items to overcome linguistic variation, rather than for referring to and teaching action sequences.

Other researchers have used semantic parsing to facilitate natural language instruction for robots. One approach learns a parser to map natural-language instructions to control language [Matuszek *et al.*, 2013]. We build on such approaches by augmenting our parser with new data in an incremental fashion from dialog. We also use world knowledge to ground natural language expressions. Other work uses restricted language and a static, hand-crafted lexicon to map natural language to action specifications [Matuszek *et al.*, 2013].

Work closest to ours presents a dialog agent used together with a knowledge base and semantic understanding component to learn new referring expressions during conversations that instruct a mobile robot [Kollar *et al.*, 2013]. They use semantic frames of actions and arguments extracted from user utterances, while we use λ-calculus meaning representations. Our agent reasons about arguments like "Mallory Morgan's office", by considering what location would satisfy it, while semantic frames instead add a lexical entry for the whole phrase explicitly mapping to the appropriate room. Our method is more flexible for multi-entity reasoning (e.g. "the person whose office is next to Mallory Morgan's office") and changes to arguments (e.g. "George Green's office"). Additionally, this work did not evaluate how agent learning affects user experience.

Our process of automatically inducing training examples from conversations is partly inspired by Artzi and Zettlemoyer [2011]. They used logs of conversations that users had with an air-travel information system to train a semantic parser for understanding user utterances. Our approach to learning is similar, but done incrementally from conversations the agent has with users, and our training procedure is integrated into a complete, interactive robot system.

3 Dialog Agent

The user first gives a command to our agent, then a dialog begins in which the agent can ask clarification questions (Figure 1). The agent maintains a belief state about the user's goal. When it is confident in this state, the dialog ends and the goal is passed on to the robot or other underlying system.

3.1 Semantic Parser

The agent produces a logical form representing what the user said. We use the University of Washington Semantic Parsing Framework (SPF) [Artzi and Zettlemoyer, 2013], a state-of-the-art system for mapping natural language to meaning representations using λ-calculus and combinatory categorial grammar (CCG). λ-calculus is a formalism for representing the meaning of lexical items. CCG [Steedman and Baldridge, 2011] tags each lexical item with a syntactic category. Given a CCG tagged utterance where categories are paired with λ-calculus expressions, a meaning representation for the whole utterance can be derived (see Figure 2).

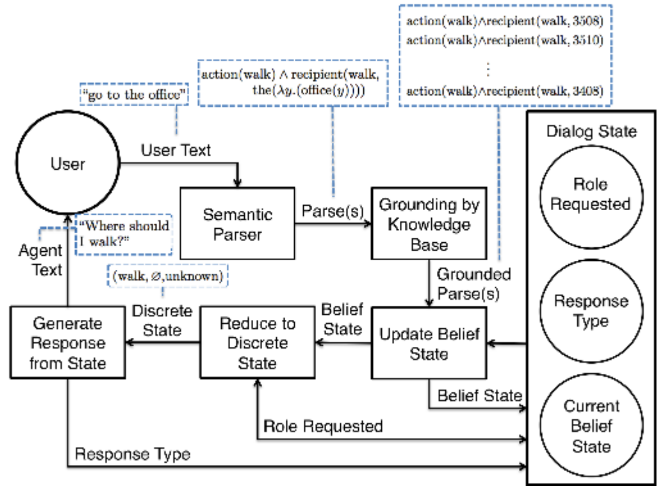

Figure 1: Dialog agent workflow. Dashed boxes show processing of user command "go to the office".

To get the system "off the ground" we initialize the parser with a small seed lexicon—pairings of lexical items with CCG categories and λ-calculus expressions—and then train it on a small set of supervised utterance/logical-form pairs. We use a seed lexicon of 105 entries (40 of which are named entities) and a training set of only 5 pairs.

3.2 Grounding by Knowledge Base

Given a λ-calculus logical form, the agent can ground some variables by querying a knowledge base of facts about the environment. Given the expression derived in Figure 2 for "Mallory Morgan's office", the agent can verify that $y = 3508$ satisfies the expression, since 3508 is an office belonging to Mallory. If there are multiple satisfying objects, all are returned, but the multiplicity decreases the agent's confidence in their correctness (Section 3.3).

3.3 Maintaining a Belief State and Responding

The agent's belief state about the user goal has three components: `action`, `patient`, and `recipient`. Each component is a histogram of confidences over possible assignments. The agent supports two actions: walking and bringing items, so the belief state for `action` is two confidence values in $[0, 1]$. `recipient` and `patient` can take values over the space of entities (people, rooms, items) in the knowledge base as well as a null value $\varnothing$. All confidences are initialized to zero when a new conversation starts.

Updating the Belief State: Multiple meaning hypotheses may be generated from a user utterance. Consider:

expression	go to the office
logical form	action(walk) $\wedge$ recipient(walk, the(λy.(office(y))))

For n offices, this logical form has n groundings producing different meanings (see Figure 1). The agent can be confident that walking is the task, but its confidence in the n meanings for `recipient` is weakened. We use a simple confidence update based on the number k of hypotheses generated to track the agent's confidence in its understanding of each

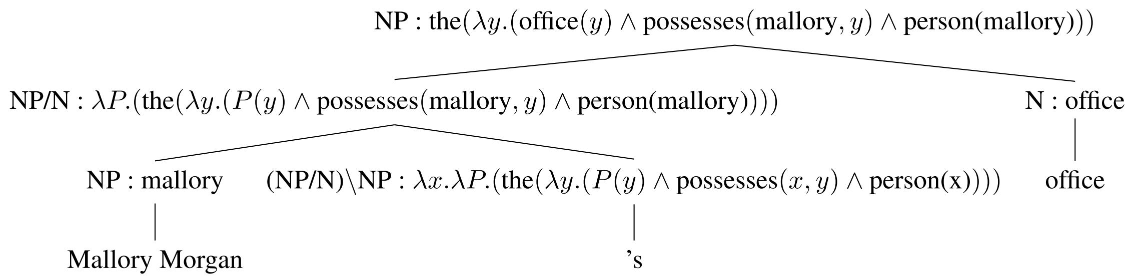

Figure 2: A CCG-driven λ-calculus parse of the expression “Mallory Morgan’s office”.

component of the request. For a user-initiative (open-ended) statement like this one, the agent updates all components of the belief state. For each candidate hypothesis $H_{i,c}$, with $0 \leq i < k$, $c \in \{\texttt{action}, \texttt{patient}, \texttt{recipient}\}$, the agent updates:

$$\text{conf}(c = H_{i,c}) \leftarrow \text{conf}(c = H_{i,c}) \left(1 - \frac{\alpha}{k}\right) + \frac{\alpha}{k}$$

Where $0 < \alpha < 1$ is the threshold of confidence above which the candidate is accepted without further clarification. The confidence in unmentioned arguments is decayed to wash out previous misunderstandings when the user has been asked to re-word the goal. For A_c, the set of all candidates of component c, $\bar{A}_c = A_c \setminus \cup_i \{H_{i,c}\}$ are unmentioned. For each $\bar{H}_{j,c} \in \bar{A}_c$, the agent updates:

$$\text{conf}(c = \bar{H}_{j,c}) \leftarrow \gamma \text{conf}(c = \bar{H}_{j,c})$$

where $0 \leq \gamma \leq 1$ is a decay parameter.

System-initiative responses are associated with a particular requested component. These can take the form of confirmations or prompts for components. For the former, user affirmation will update the confidence of all mentioned values to 1. For the latter, the positive and negative updates described above operate only on the requested component.

Responding: The agent uses a static dialog policy π operating over a discrete set of states composed of `action`, `patient`, `recipient` tuples together with the role to be clarified. The agent’s continuous belief state S is reduced to a discrete state S' by considering the top candidate arguments T_c for each component c:

$$T_c = \text{argmax}_{t \in A_c} (\text{conf}(c = t))$$

Each component c of S' is selected by choosing either T_c or “unknown” with probability conf($\mathrm{c} = T_c$). The component c with the minimum confidence is chosen as the role to request. If “unknown” is chosen for every component, the role requested is “all”. If “unknown” is chosen for no component, the role requested is “confirmation”. Some policy responses are given in Table 1. If each of the confidence values inspected during this process exceeds α, the conversation concludes. In all experiments, parameters $\alpha = 0.95, \gamma = 0.5$ were used.

3.4 Learning from Conversations

The parser uses a template-based lexical generation procedure (GENLEX), described fully in [Zettlemoyer and Collins, 2005; 2007], to add new lexical items with CCG and logical forms derived from existing entries. For each utterance paired with a logical form seen during training, GENLEX generates lexical items (utterance n-grams) and pairs them with all n-gram combinations of logical constants. Pairs that lead to successful parses are added to the parser’s lexicon. Training also re-weights the parser parameters, affecting the likelihoods of particular assignments of lexical items to logical forms. So, if the parser internalizes an incorrect pair, there will be user misunderstandings initially, but the pair will be discredited as correct contexts are seen.

Our agent induces parsing training examples from conversations with users to learn new lexical items. It uses dialog conclusions and explicit confirmations from users as supervision. The semantic parser in Figure 3 does not know the misspelling “calender”, the word “planner”, or number “5”. When the user requests “item in slot 5” be delivered, it only confidently detects the `action`, “bring”, of the user’s goal. The `recipient`, “Dave Daniel”, is clarified by a system-initiative question. When the agent asks for confirmation of the `action`, the user does not deny it, increasing the agent’s confidence. While clarifying the `patient`, the user implicitly provides evidence that “calender”, “planner”, and “calendar” are the same. When two or more phrases are used in the same sub-dialog to clarify an argument, the eventual logical form selected is paired with the earlier surface forms for retraining.

User-initiative responses generate similar alignments. One users’ conversation began “please report to room 3418”, which the agent could not parse because of the new word “report”. The agent understood the re-worded request “go to room 3418”, and the former sentence was paired with the logical form of this latter for training. When the GENLEX procedure explored possible semantic meanings for “report”, it found a valid parse with the meaning of “go”, “S/PP : λP.(action(walk) $\wedge$ P(walk))”, and added it to the parser’s lexicon. This meaning says that “report” should be followed by a prepositional phrase specifying a target for the walking `action`.

4 Overview of Experiments

We evaluated the learning agent in two contexts. We used Mechanical Turk to gather data from many diverse users asked to give the agent goals for an office environment. These users interacted with the agent through a web browser, but

Table 1: Representative subset of our policy π for mapping discrete states S' to actions (questions to ask the user).

S'		$\pi(S')$	
(action,patient,recipient)	**Role Request**	**Text**	**Initiative**
(unknown,unknown,unknown)	all	Sorry I couldn't understand that. Could you reword your original request?	user
(unknown,$T_{\texttt{patient}}$,$T_{\texttt{recipient}}$)	action	What action did you want me to take involving $T_{\texttt{patient}}$ and $T_{\texttt{recipient}}$?	system
(walk, $\varnothing$,unknown)	recipient	Where should I walk?	system
(bring,unknown,$T_{\texttt{recipient}}$)	patient	What should I bring to $T_{\texttt{recipient}}$?	system
(walk, $\varnothing$, $T_{\texttt{recipient}}$)	confirmation	You want me to walk to $T_{\texttt{recipient}}$?	system
(bring, $T_{\texttt{patient}}$, $T_{\texttt{recipient}}$)	confirmation	You want me to bring $T_{\texttt{patient}}$ to $T_{\texttt{recipient}}$?	system

user expectations, frustrations, and lexical choices with a web browser versus a physical robot will likely differ. Thus, we also implemented an interface for the agent on a Segway-based robot platform (Segbot) operating on a floor of our university's computer science building.

We split the possible task goals into train and test sets. In both contexts, users performed a *navigation* (send robot to a room) and a *delivery* (have an item delivered to a person) task. For the 10 possible navigation goals (10 rooms), we randomly selected 2 for testing. For the 50 possible delivery goals (10 people $\times$ 5 items), we randomly selected 10 for testing (80%/20% train/test split). The test goals for Mechanical Turk and the Segbot were the same, except in the former we anonymized the names of the people on our building's floor.

We ended all user sessions with a survey: "The tasks were easy to understand" (*Tasks Easy*); "The robot understood me" (*Understood*); and "The robot frustrated me" (*Frustrated*). For the Segbot experiment, we also prompted "I would use the robot to find a place unfamiliar to me in the building" (*Use Navigation*) and "I would use the robot to get items for myself or others" (*Use Delivery*). Users answered on a 5-point Likert scale: "Strongly Disagree"(0), "Somewhat Disagree"(1), "Neutral"(2), "Somewhat Agree"(3), "Strongly Agree"(4). Users could also provide open comments.

5 Mechanical Turk Experiments

The web interface shown in Figure 3 was used to test the agent with many users through Mechanical Turk.

5.1 Methodology

Each user participated in *navigation*, *delivery*, and *validation* tasks, then filled out the survey. We performed incremental learning in batches to facilitate simultaneous user access. We assigned roughly half of users to the test condition and the other half to the train condition per batch. After gathering train and test results from a batch, we retrained the parser using the train conversation data. We repeated this for 3 batches of users, then we gathered results from a final testing batch in which there was no need to gather more training data. We used user conversations for retraining only when they achieved correct goals.

Navigation: Users were asked to send the robot to a random room from the appropriate train or test goals with the prompt "[person] needs the robot. Send it to the office where [s]he works". The referring expression for each person was chosen from: full names, first names, nicknames, and titles ("Dr. Parker", "the Director"). In this task, the corresponding office number was listed next to each name, and the "items available" were not shown.

Delivery: Users were asked to tell the robot to assist a person with the prompt "[person] wants the item in slot [number]". The $(person, item)$ pairs were selected at random from the appropriate train or test goals. To avoid linguistic priming, the items were given pictorially (Figure 3).

Validation: To detect users who were not taking the tasks seriously, we selected a random office and asked them to "Give only the first and last name of the person in office number [number]". Incorrect responses were hand-inspected and either validated or marked invalid. Validated users made innocuous errors like misspellings. Only 17 of 353 users were marked invalid after hand-inspection and were left out of further analysis.

5.2 Results

For each train/test condition, we gathered responses from an average of 48 users per batch. Figure 4 (Left) shows the mean survey-question responses across test batches. We used an unpaired Welch's two-tailed t-test to determine whether these means differed significantly. By batch 2, users felt that the agent understood them more than in batch 0. By batch 3, they felt that it frustrated them less. The dialog agent became more understandable and likable as a result of the semantic parser's learning, even though it had never seen the test-batch users' goals.

To determine whether learning reduced the number of utterances (turns) a user had to provide for the system to understand their goal, we counted user turns for dialogs where the user and agent agreed on the correct goal (Figure 4 (Right)). Learning successfully reduced the turns needed to understand multi-argument delivery goals.

With respect to users' free-form feedback, in testing batch 0, several enjoyed their conversations ("This was fun!! Wish it were longer!"). Several also commented on the small initial lexicon ("It was fun to try and learn how to talk to the robot in a way it would understand"). The responses by testing batch 3 had similarly excited-sounding users ("I had so much fun doing this hit!"). At least one user commented on the lexical variation they observed ("The robot fixed my grammatical error when I misspelled 'calender' Which was neat"). In ad-

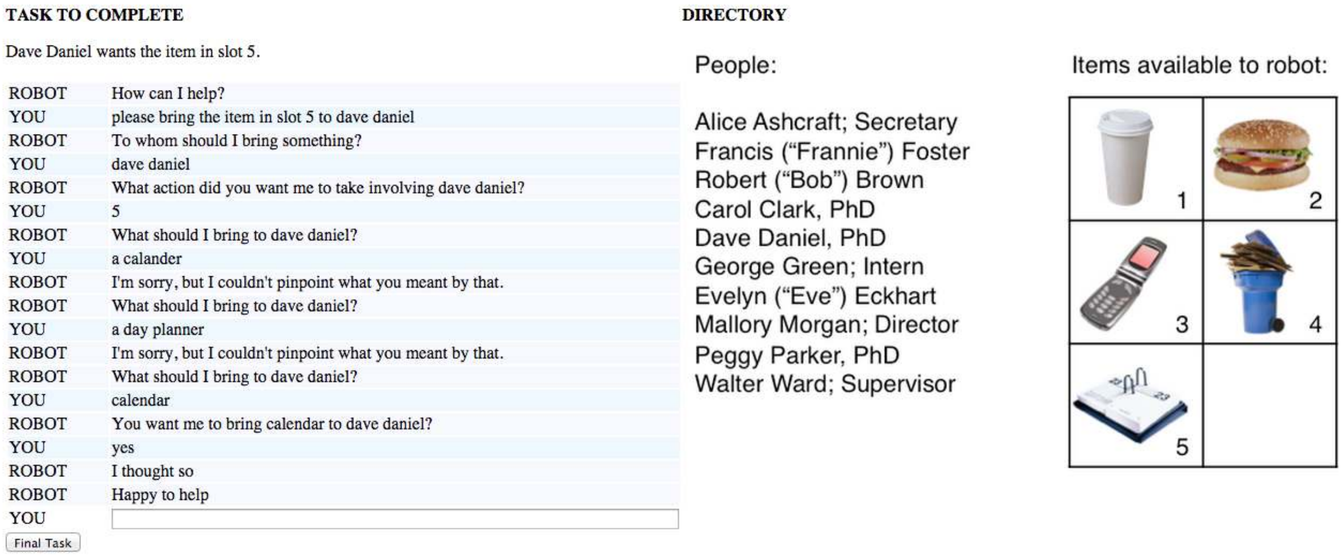

Figure 3: The Mechanical Turk interface for the delivery task. This abridged conversation is from a Turker in training batch 0, when the system had access to only the seed lexicon. Because of this conversation, the agent learned that "calender" and "planner" mean "calendar" during retraining.

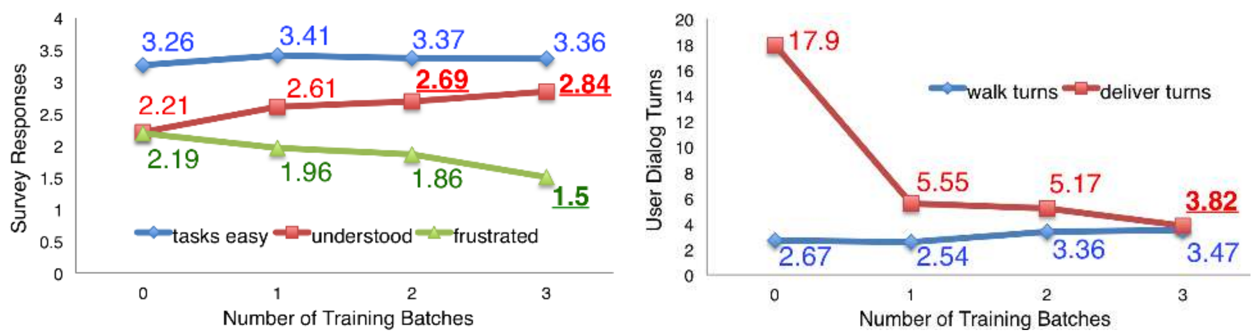

Figure 4: **Left**: Average Mechanical Turk survey responses across the four test batches. **Right**: Mean user turns in Mechanical Turk dialogs where the correct goal was reached. Means in underlined bold differ significantly ($p < 0.05$) from the batch 0 mean.

dition to learning misspelling corrections and new referring expressions, the agent learned to parse things like "item in slot n" by matching n to the corresponding item and collapsing the whole phrase to this meaning.

6 Segbot Experiments

The agent was integrated into a Segway-based robot platform (Segbot) as shown in Figure 5 (Left) using the Robot Operating System (ROS) [Quigley *et al.*, 2009].

6.1 Implementation

The robot architecture is shown in Figure 5 (Right). Users interacted with the agent through a graphical user interface by typing in natural language. The agent generated queries to a symbolic planner formalized using action language $\mathcal{BC}$ [Lee *et al.*, 2013] from user goals. Action languages are used for representing and reasoning with the preconditions, effects, and executability of actions, and $\mathcal{BC}$ is good at reasoning with domain knowledge. The sensor readings were converted to logical facts provided to the symbolic planner. For instance, we used laser sensors to detect whether office doors were open. The Segbot learned action costs from experience using an existing approach [Khandelwal *et al.*, 2014], and the symbolic planner generated lowest-cost plans. The action executor used a manually-created semantic map to translate symbolic actions into path-planner executions. We used existing ROS packages for path planning (e.g. A* search for global path planning and Elastic Band for local path planning). The sensor readings from the RGB-D camera (Kinect), laser, and sonar array were projected onto a 2D costmap so that the robot could safely avoid obstacles such as high tables and glass windows.

6.2 Methodology

For testing, users were given one goal from the navigation and delivery tasks, then filled out the survey. The task prompts

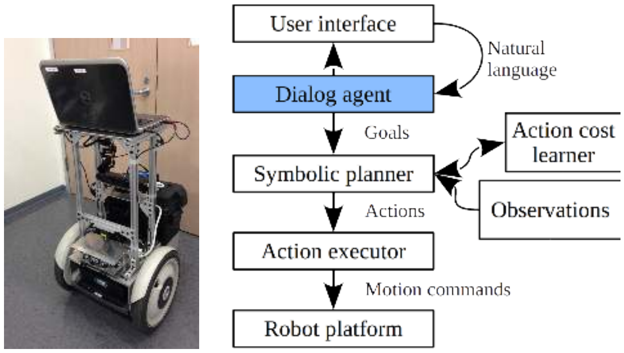

Figure 5: **Left:** Robot platform (Segbot) used in experiments. **Right:** Segbot architecture, implemented using Robot Operating System (ROS).

included the directory panels used in the Mechanical Turk experiments pairing names and office numbers and showing items available to the robot for delivery (Figure 3).

We evaluated our agent's initial performance by giving 10 users one of each of these goals (so each delivery test goal was seen once and each navigation test goal was seen 5 times). Users were allowed to skip goals they felt they could not convey. We refer to this group as `Init Test`.

We then allowed the agent to perform incremental learning for four days in our office space. Students working here were encouraged to chat with it, but were not instructed on how to do so beyond a panel displaying the directory information and a brief prompt saying the robot could only perform "navigation and delivery tasks". Users in test conditions did not interact with the robot during training. After understanding and carrying out a goal, the robot prompted the user for whether the actions taken were correct. If they answered "yes" and the goal was not in the test set, the agent retrained its semantic parser with new training examples aligned from the conversation. View a video demonstrating the learning process on the Segbot at: `https://youtu.be/FL9IhJQOzb8`.

We evaluated the retrained agent as before. The same testing goal pairs were used with 10 new users. We refer to this latter set as `Trained Test`.

6.3 Results

During training, the robot understood and carried out 35 goals, learning incrementally from these conversations. Table 2 compares the survey responses of users and the number of goals users completed of each task type in the `Init Test` and `Trained Test` groups. Because only two users completed delivery goals in `Init Test`, we use the proportion of users having completed goals in each task, rather than conversation length, as a metric for dialog efficiency. For navigation goals, `Init Test` had an average dialog length of 3.89, slightly longer than the 3.33 for `Train Test`.

We note that there is significant improvement in user perception of the robot's understanding, and trends towards less user frustration and higher delivery-goal correctness. Though users did not significantly favor using the robot for tasks after training, several users in both groups commented that they

Table 2: Average Segbot survey responses from the two test groups and the proportion of task goals completed. Means in bold differ significantly ($p < 0.05$). Means in italics trend different ($p < 0.1$).

	`Init Test`	`Trained Test`
Survey Question	Likert [0-4]	
Tasks Easy	3.8	3.7
Robot Understood	1.6	**2.9**
Robot Frustrated	2.5	*1.5*
Use Navigation	2.8	2.5
Use Delivery	1.6	2.5
Goals Completed	Percent	
Navigation	90	90
Delivery	20	*60*

would not use guidance only because the Segbot moved too slowly.

7 Conclusions and Future Work

We implemented an agent that expands its natural language understanding incrementally from conversations with users by combining semantic parsing and dialog management. We demonstrated that this learning yields significant improvements in user experience and dialog efficiency through Mechanical Turk experiments with hundreds of users. A proof-of-concept experiment on a Segbot platform showed similar improvements when learning was restricted to natural conversations the agent had over a few days' time.

This work provides initial steps towards expanding natural-language understanding for robot commands using natural conversations with users as training data. Our agent improves its language understanding without requiring a large corpus of annotated data.

We intend to replace our static dialog policy with a POMDP-based policy [Young *et al.*, 2013] that considers the continuous belief state about the user goal. Incremental learning will then involve updating the dialog policy through reinforcement learning based on parser confidence and conversation success. We will also explore whether our approach can automatically learn to correct consistent speech recognition errors. As the robot platform gains access to more tasks, such as manipulation of items, doors, and light-switches via an arm attachment, we will scale the agent to learn the language users employ in that larger goal space. We also plan to add agent perception, so that some predicates can be associated with perceptual classifiers [Matuszek *et al.*, 2012], and new predicates can be discovered for new words.

8 Acknowledgments

We thank the anonymous reviewers for their feedback. A portion of this work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (CNS-1330072, CNS-1305287), ONR (21C184-01), and AFOSR (FA8750-14-1-0070, FA9550-14-1-0087).

References

[Artzi and Zettlemoyer, 2013] Yoav Artzi and Luke Zettlemoyer. UW SPF: The University of Washington Semantic Parsing Framework, 2013.

[Cantrell *et al.*, 2012] Rehj Cantrell, Kartik Talamadupula, Paul Schermerhorn, J. Benton, Subbarao Kambhampati, and Matthias Scheutz. Tell me when and why to do it!: Run-time planner model updates via natural language instruction. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 471–478, New York, NY, USA, 2012. ACM.

[Eberhard *et al.*, 2010] Kathleen Eberhard, Hannele Nicholson, Sandra Kübler, Susan Gunderson, and Matthias Scheutz. The Indiana "Cooperative Remote Search Task" (CReST) Corpus. In *Proceedings of the Seventh Language Resources and Evaluation Conference (LREC)*, 2010.

[Khandelwal *et al.*, 2014] Piyush Khandelwal, Fangkai Yang, Matteo Leonetti, Vladimir Lifschitz, and Peter Stone. Planning in Action Language $\mathcal{BC}$ while Learning Action Costs for Mobile Robots. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2014.

[Klein and Manning, 2003] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL) - Volume 1*, pages 423–430, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[Kollar *et al.*, 2013] Thomas Kollar, Vittorio Perera, Daniele Nardi, and Manuela Veloso. Learning environmental knowledge from task-based human-robot dialog. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4309, Karlsruhe, 2013.

[Lee *et al.*, 2013] Joohyung Lee, Vladimir Lifschitz, and Fangkai Yang. Action Language $\mathcal{BC}$: A Preliminary Report. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

[Matuszek *et al.*, 2012] Cynthia Matuszek, Nicholas Fitzgerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. A joint model of language and perception for grounded attribute learning. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1671–1678, New York, NY, USA, 2012. ACM.

[Matuszek *et al.*, 2013] Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In *Experimental Robotics*, pages 403–415. Springer International Publishing, 2013.

[Misra *et al.*, 2014] Dipendra Kumar Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me Dave: Context-sensitive grounding of natural language to manipulation instructions. In *Proceedings of Robotics: Science and Systems (RSS)*, Berkeley, USA, July 2014.

[Quigley *et al.*, 2009] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. ROS: an open-source robot operating system. In *Open Source Softare in Robotics Workshop at the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[She *et al.*, 2014] Lanbo She, Shaohua Yang, Yu Cheng, Yunyi Jia, Joyce Chai, and Ning Xi. Back to the blocks world: Learning new actions through situated human-robot dialogue. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 89–97, Philadelphia, PA, U.S.A., June 2014. Association for Computational Linguistics.

[Steedman and Baldridge, 2011] Mark Steedman and Jason Baldridge. Combinatory categorial grammar. In Robert Borsley and Kersti Borjars, editors, *Non-Transformational Syntax: Formal and Explicit Models of Grammar*. Wiley-Blackwell, 2011.

[Tellex *et al.*, 2011] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R. Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2011.

[Young *et al.*, 2013] Steve Young, Milica Gasic, Blaise Thomson, and Jason D Williams. POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[Zettlemoyer and Collins, 2005] Luke S. Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the 21st Conference on Uncertainty in AI (UAI)*, pages 658–666, 2005.

[Zettlemoyer and Collins, 2007] Luke S. Zettlemoyer and Michael Collins. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687, 2007.