Journal of Artificial Intelligence Research 67 (2020) 327-374 Submitted 05/2019; published 02/2020

Jointly Improving Parsing and Perception for Natural Language Commands through Human-Robot Dialog

Jesse Thomason JDTHO@CS.WASHINGTON.EDU
University of Washington
Paul G. Allen School of Computer Science & Engineering

Aishwarya Padmakumar AISH@CS.UTEXAS.EDU
University of Texas at Austin
Department of Computer Science

Jivko Sinapov JIVKO.SINAPOV@TUFTS.EDU
Tufts University
Department of Computer Science

Nick Walker NSWALKER@CS.WASHINGTON.EDU
University of Washington
Paul G. Allen School of Computer Science & Engineering

Yuqian Jiang JIANGYUQIAN@UTEXAS.EDU
Harel Yedidsion HAREL@CS.UTEXAS.EDU
Justin Hart HART@CS.UTEXAS.EDU
Peter Stone PSTONE@CS.UTEXAS.EDU
Raymond J. Mooney MOONEY@CS.UTEXAS.EDU
University of Texas at Austin
Department of Computer Science

Abstract

In this work, we present methods for using human-robot dialog to improve language understanding for a mobile robot agent. The agent parses natural language to underlying semantic meanings and uses robotic sensors to create multi-modal models of perceptual concepts like *red* and *heavy*. The agent can be used for showing navigation routes, delivering objects to people, and relocating objects from one location to another. We use dialog clarification questions both to understand commands and to generate additional parsing training data. The agent employs opportunistic active learning to select questions about how words relate to objects, improving its understanding of perceptual concepts. We evaluated this agent on Amazon Mechanical Turk. After training on data induced from conversations, the agent reduced the number of dialog questions it asked while receiving higher usability ratings. Additionally, we demonstrated the agent on a robotic platform, where it learned new perceptual concepts on the fly while completing a real-world task.

1. Introduction

Humans use natural language to articulate their thoughts and intentions. As robots are deployed across diverse human environments, such as homes, offices, and hospitals, the need for smooth human-robot communication is growing. The language we use to discuss these

©2020 AI Access Foundation. All rights reserved.

environments varies, with domain-specific words and affordances in each (e.g., *turn on the living room lights*; *move the pallet a few feet to the north*; *notify me if the patient's condition changes*). This variety makes pre-programming robots with fixed language understanding components a brittle solution. We propose methods for robots to leverage human speaking partners as a source of additional learning signals for language understanding.

Human-robot dialog can involve understanding semantically complex commands. Consider the command:

Move the light mug from Bob's office to the west, middle pod. (1)

Commands can be translated into machine-readable meaning representations (also called *semantic forms*). Given a semantic form, a robot can use world knowledge and perception to resolve references to the real world. For example, one semantic form corresponding to the preceding command is:

$$\begin{aligned}\text{relocate}(&\text{the}(\lambda x.(\text{light}_{\text{weight}}(x) \wedge \text{mug}(x))),\\ &\text{the}(\lambda y.(\text{office}(y) \wedge \text{owns}(\text{robert}, y))),\\ &\text{the}(\lambda z.(\text{west}(z) \wedge \text{middle}(z) \wedge \text{pod}(z)))).\end{aligned} \tag{2}$$

The λ expressions denote variables to be instantiated conditioned on the environment (e.g., what place is both an office and is owned by `robert`). *Grounding* these variables is another stage required for understanding, which we discuss in Section 3.3.

Translating human utterances to semantic forms helps handle the synonymy of commands and words (e.g., *Bob* for *Robert*), compositionality (e.g., *Bob's office*, *the light mug*), and ambiguity (e.g., *light* in weight versus *light* in color). This work ameliorates the annotation efforts often required to build a *semantic parser* to translate between language and semantic forms. We expand on past work (Thomason, Zhang, Mooney, & Stone, 2015) to induce parser training data directly from human-robot conversations.

To communicate about a shared environment, robots must gather and maintain world knowledge through perception. Some world knowledge can be modeled as static, such as the layout of a building. This information can be created by humans and used for language understanding. For example, in (1), the parse of *Bob's office* can be grounded against such knowledge to find the room satisfying these constraints. Other world knowledge is perceptual, such as whether an object is a *mug*. A service robot in a human environment needs both types of knowledge to understand and respond to human requests.

Labeled examples allow a classifier to tie *mug* with properties of objects in the world. Exhaustively obtaining these labels is a time-consuming human effort. Consider the word *heavy*, which may require a person to physically lift each object to decide the label. Instead, we propose to extract these labels from natural human-robot conversations. We use *opportunistic active learning* to enable a robot to ask questions about how concept words apply to objects that are *local* to the human and robot. If a human requests *an empty cup*, the robot can point to a local object whose emptiness it is unsure of and ask for the human's opinion, improving the concept model for *empty* with this additional labeled example. We previously found that humans are undeterred by these types of questions, even if they are not related to the requested object (Thomason et al., 2017).

In this paper, we present a dialog agent that jointly improves parsing and perception on a robotic system for natural language commands. We expand on our earlier work (Thomason et al., 2019), describe the agent and its components in full, add a dialog recovery strategy, and evaluate the agent on Mechanical Turk. Figure 1 shows the Mechanical Turk interface, an example command, and the beginning of a human-agent dialog. The agent asks fewer clarification questions that require tedious feedback from users (e.g., scrolling through a list of possible options) after learning. Additionally, users rate the agent more favorably for potential use in real-world tasks after learning. To demonstrate flexibility in understanding these non-visual predicates, we also implement the agent on a physical robot (Khandelwal et al., 2017) and use it to drive human-robot dialog.[1] The embodied agent is able to learn the meaning of a new word, *rattling*, on the fly to complete an object relocation task.

Our implementation makes several key assumptions:

A.1 Robot actions can be broken down into tuples of semantic roles (e.g., *action*, *patient*).

A.2 The language used in the domain is broader than the annotated seed data.

A.3 The user is cooperative, answering all questions on-topic and truthfully.

A.4 User commands always specify a discrete action that the robot can perform.

A.5 The robot has a closed world of domain referents (e.g., objects).

A.6 The domain of perceptual concepts is open, but concepts are independent (e.g., a *red mug* means something describable as both *red* and as a *mug*), categorical (e.g., *heavy* is not graded), and uniquely identified by a word (e.g., *light* is assumed as either just lightweight or just light in color).

In the remainder of this article, we discuss background information and related work in the space of language understanding for embodied robots (Section 2), present the learning agent (Section 3), discuss the Mechanical Turk evaluation and robot demonstration (Section 4), and summarize our contributions and discuss directions for future work (Section 5).

Our approach has three primary contributions. We:

C.1 **use human-robot dialog** to identify user requests and to generate training data to

C.2 improve the dialog agent's **semantic parsing abilities** (e.g., learn new referring expressions and syntactic constructions), and

C.3 improve the dialog agent's **perceptual grounding abilities** (e.g., how words like *red* and *rattling* relate to physical object properties).

2. Background and Related Work

We now discuss existing work on instructing robots through natural language, a task our agent performs using both semantic parsing and perception. We then describe relevant work on learning semantic parsers, especially from conversations. Finally, we discuss language grounding in machine perception and in human-robot interactions.

1. The demonstration video can be viewed at `https://youtu.be/PbOfteZ_CJc`.

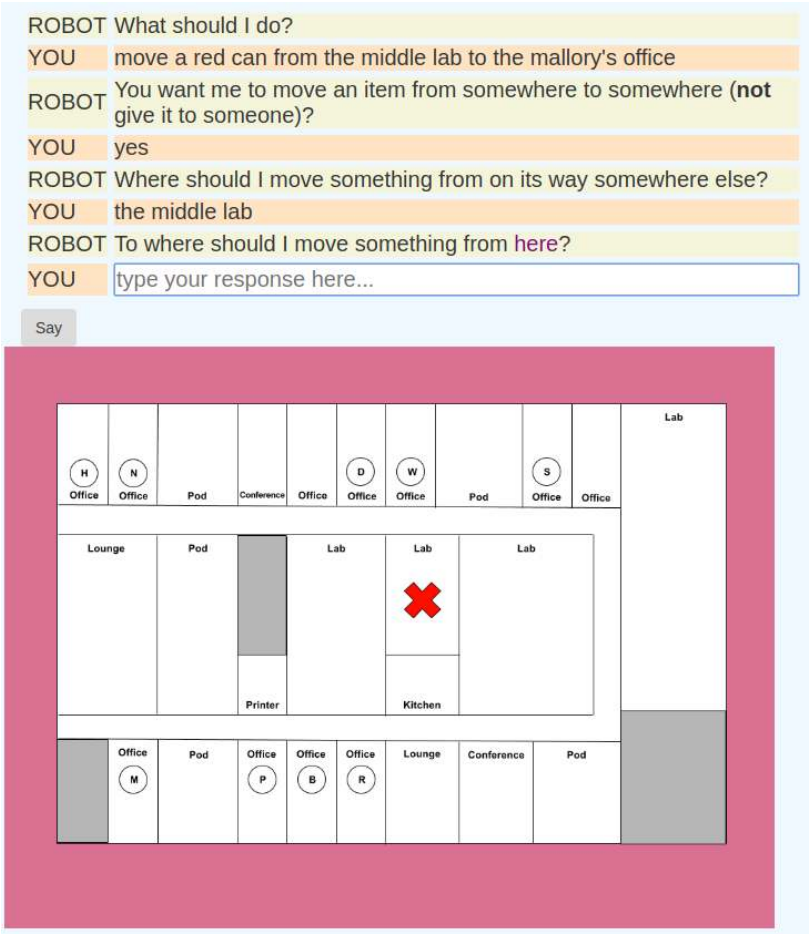


Figure 1: The Mechanical Turk web interface. The user types a command to the learning agent, which replies with questions to clarify the command until the user confirms understanding. In this conversation, the agent has just asked a confirmation question—whether it has understood the correct location of the *red can*. The purple text *here* refers to the location marked by an X in the map below the conversation window.

2.1 Instructing Robots using Natural Language

Instructing robots through natural language is useful for human-robot cooperation. Language understanding can enable robots (Kollar, Tellex, Roy, & Roy, 2010; Matuszek, Herbst, Zettlemoyer, & Fox, 2012b, 2013) and simulated agents (Wang, Xiong, Wang, & Yang Wang, 2018; Shah, Fiser, Faust, Kew, & Hakkani-Tur, 2018) to navigate in new, even photorealistic (Anderson et al., 2018) environments. In this work, we use weak supervision to improve language understanding based on interactions with humans rather than learning from a fixed corpus of commands paired with behaviors (Artzi & Zettlemoyer, 2013b; Thomason et al., 2015).

Grounding referents probabilistically can be based on sensor data and human language (Vanzo, Croce, Bastianelli, Basili, & Nardi, 2019; Walter, Hemachandra, Homberg,

Tellex, & Teller, 2013) in addition to static information (Mohan, Mininger, & Laird, 2013). Generalized grounding graphs facilitate both understanding and the generation of language requests regarding the shared environment (Tellex, Knepper, Li, Rus, & Roy, 2014). Extensions of this framework can be used to memorize new semantic referents (Paul, Barbu, Felshin, Katz, & Roy, 2017), to reason about abstract sets (Paul, Arkin, Roy, & Howard, 2016), and to function efficiently as the number of objects and properties grows (Patki, Daniele, Walter, & Howard, 2019). Recent work also ties language to planning infrastructure (Skoviera et al., 2018; Chai et al., 2018; Nyga et al., 2018) and rewards (Williams, Gopalan, Rhee, & Tellex, 2018). Combining threads from several of these lines of work, we assume a knowledge base for information (e.g., the environment layout). We then ground referring expressions to objects in the real world using learned perceptual classifiers and represent groundings as values that fill semantic roles (A.1).

Some works utilize object discovery or environmental exploration. However, we assume a closed world with a fixed set of referents (A.5) for which we can learn additional referring expressions (A.2) through semantic parsing.

2.2 Learning Semantic Parsers

A *semantic parser* maps words to semantic forms, often defined by some ontology and dependent on a lexicon connecting words to meanings. While some semantic parsing work does not rely on a fixed ontology, we assume a structured representation of the world extensible through learning (A.1 and A.2), and we consider only semantic forms defined by an ontology throughout this article. A *semantic parse* is a meaning representation comprised of ontological predicates and λ expressions. A λ expression abstracts meaning over sets of ontological constants and predicates. The meaning of *go to alice's office* could be represented as:

$$\text{go}(\text{the}(\lambda x.(\text{office}(x) \wedge \text{owns}(\text{alice}, x))))$$

In this case, *the* will pick out a unique constant x that is both an `office` and belongs to `alice`. This constant is the argument for the `go` command.

We follow previous work in parser learning (Lee, Lewis, & Zettlemoyer, 2016; Lewis, Lee, & Zettlemoyer, 2016; Misra & Artzi, 2016) and learn a parser based on *combinatory categorial grammar* (CCG) (Steedman & Baldridge, 2011). The CCG formalism adds a syntactic category to each token of the input sequence t and combines those categories to build a syntax tree. Syntactic categories are typically a small set, like N (noun) and NP (noun phrase), together with compositional entries, like PP/NP (a prepositional phrase formed by consuming a noun phrase to its right). The input lexicon L contains entries mapping word tokens to both meaning representations and CCG syntax categories. Meanings in this representation can be composed according to the hierarchy established by a CCG parse tree to form a semantic parse for a token sequence (Figure 2).

Given token sequences paired with their semantic parse trees, a statistical parser can be trained to produce parses for novel sequences (Liang & Potts, 2015; Berant, Chou, Frostig, & Liang, 2013). Neural methods perform this translation using sequence-to-sequence (Konstas, Iyer, Yatskar, Choi, & Zettlemoyer, 2017; Jia & Liang, 2016) or sequence-to-tree (Dong & Lapata, 2016) neural networks, often in a semi-supervised fasion (Kočiskỳ et al., 2016). However, annotating token sequences with entire parse trees is costly, and we instead train

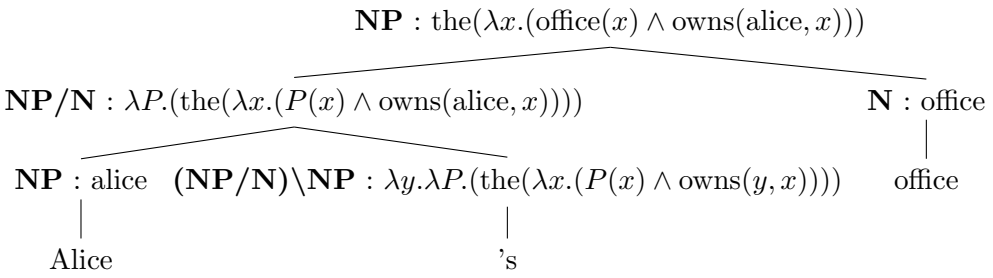


Figure 2: A CCG-driven λ-calculus parse of the expression *Alice's office*.

with only the semantic form (Liang, Jordan, & Klein, 2011). Some works incorporate ontology matching (Kwiatkowski, Choi, Artzi, & Zettlemoyer, 2013); instead we have an open domain of predicates and closed domain of objects (A.5).

Annotating natural language with semantic forms is an expensive human effort. Consequently, some work has focused on overcoming data sparsity during training. For example, identifying incorrect parses using unskilled annotators enables active learning to rectify mistakes using domain experts (Iyer, Konstas, Cheung, Krishnamurthy, & Zettlemoyer, 2017). Other work directly queries human users to resolve ambiguities (He, Michael, Lewis, & Zettlemoyer, 2016b) or induces training examples automatically from existing conversational data (Artzi & Zettlemoyer, 2011).

In our past work, we built on these ideas to gather training examples for a semantic parser from human-robot dialog (Thomason et al., 2015). Here, we train a parser using examples induced from human-robot conversations under the assumption that users are cooperative (A.3 and A.4). We extend this dialog strategy to additionally learn language grounding in robot perception.

2.3 Language Grounding with Machine Perception

Mapping from a referring expression (e.g., *the blue cup*) to an object referent in the world is an example of the *symbol grounding problem* (Harnad, 1990). Symbol grounding connects internal representations of information in a machine to world knowledge and sensory perception. *Grounded language learning* bridges symbols with natural language.

Early work coupled vision with speech descriptions of objects to learn grounded semantics (Roy & Pentland, 2002). Many approaches learn perceptual classifiers for words given some pairing of human descriptions and labeled visual scenes (A. Lazaridou & Baroni., 2014; Sun, Bo, & Fox, 2013). Some approaches additionally incorporate language modeling or semantics into the learning phase (Fahnestock, Patki, & Howard, 2019; Paxton, Bisk, Thomason, Byravan, & Fox, 2019; Bisk, Shih, Choi, & Marcu, 2018; Pillai & Matuszek, 2018; FitzGerald, Artzi, & Zettlemoyer, 2013; Krishnamurthy & Kollar, 2013; Zitnick & Parikh, 2013; Matuszek, FitzGerald, Zettlemoyer, Bo, & Fox, 2012a). In this article, our agent learns perceptual classifiers given pairings of referring expressions (induced from clarification dialogs) and object referents, together with labels between individual concept words (e.g., *blue*) and objects obtained through active learning during dialog.

Neural methods can bypass explicit symbol grounding for bridging language and visual perception (Alomari et al., 2017; Hu et al., 2016; Mao et al., 2015). These end-to-end neural approaches can require more data than is realistic for a deployed robot in a situated environment, though they work well in simulated, toy domains (Bisk, Yuret, & Marcu, 2016). For example, when object representations involve physical exploration by grasping and lifting to sense shape and weight, large scale data collection is difficult. Here, we bootstrap visual object representations with pre-trained neural networks for ImageNet classification (He, Zhang, Ren, & Sun, 2016a) and learn to ground perceptual concepts across modalities with less data-hungry methods.

Humans associate words with multiple sensory modalities (Lynott & Connell, 2009). Some prior work bridges language to audio (Kiela & Clark, 2015) and haptic (Chu et al., 2013) signals, sometimes with vision as well (Gao, Hendricks, Kuchenbecker, & Darrell, 2016). Haptic and proprioceptive feedback from a robot arm can predict object properties like weight, height, and width (Sinapov, Khante, Svetlik, & Stone, 2016). We consider visual, audio, and haptic signals when grounding concept words to object properties.

In our past work, we introduced such multi-modal perception for a robotic system that learned from conversations with humans (Thomason, Sinapov, Svetlik, Stone, & Mooney, 2016). Here, we extend this learning agent—capable of connecting these kinds of multi-modal perceptual words like *heavy* to real-world object properties—to the context of task-oriented command understanding. We make the simplifying assumptions of a closed world of objects (e.g., feature representations are available for all objects) (A.5) and concept words that are independent, categorical, and unique (A.6).

2.4 Language Grounding with Human-Robot Interaction

Machine perception can be improved through human-robot interaction. Past work has focused on solving the symbol grounding problem for situated robots by leveraging their interactions with the humans they are working to understand.

Using human descriptions of objects together with deictic hand gestures, researchers can train a grounding system for identifying referent objects (Perera & Allen, 2013; Matuszek, Bo, Zettlemoyer, & Fox, 2014; Whitney, Eldon, Oberlin, & Tellex, 2016; Whitney, Rosen, MacGlashan, Wong, & Tellex, 2017; Pizzuto, Hospedales, Capirci, & Cangelosi, 2019). Other researchers have focused on learning categorical properties of objects (*red*) together with relational (*taller*) and differentiating (*differ by weight*) properties of objects by exploring them with a robotic arm (Sinapov, Schenck, & Stoytchev, 2014b). These kinds of relations can also be orthogonally learned from text data (Forbes & Choi, 2017). We assume all perceptual concepts are categorical in nature (A.6), and we use dialogs with humans, rather than corpora, to both learn new concept models and refine existing ones.

Perceptual grounding data can be gathered via interaction with a human interlocutor. This setting affords opportunities for smart interactions, such as robots asking questions targeting weaknesses in understanding (Thomason et al., 2016). Early work on learning to ground object attributes and names via dialog used a *20 Questions*-style game (Vogel, Raghunathan, & Jurafsky, 2010). Other research focused on acquiring perceptual understandings through a command-based, rather than a game-based environment (Dindo & Zambuto, 2010). Researchers have carried this idea to more complete systems with both

perceptual grounding and action learning capabilities for identifying and manipulating objects (Mohan, Mininger, Kirk, & Laird, 2012) and for both goal-oriented and procedural actions (Mininger & Laird, 2018). These methods assume an underlying perceptual categorization for which humans can provide details. For example, a human user may specify that unknown word *orange* is a *color*; in our work, the agent learns these abstractions automatically. This enables our system to handle different types of adjectives (e.g., weights, heights, contents, etc.) without an explicit hierarchy.

Researchers have framed learning attribute classifiers for objects through human-agent dialog (Yang, Lu, Lee, Batra, & Parikh, 2018; Vanzo, Part, Yu, Nardi, & Lemon, 2018), including in a game setting (Parde et al., 2015). In another game-like approach, users offered commands and selected a correct postcondition world from the set of worlds resulting from different actions in response to the command (Wang, Liang, & Manning, 2017). In our past work, we bootstrapped a perception system using an interactive *I Spy* game (Thomason et al., 2016) and in conversations during an object identification task (Thomason et al., 2017). We introduce continual learning to the perceptual component of a grounded dialog agent by engaging in perception-relevant sub-dialogs during command-oriented conversations with humans (Thomason et al., 2019). In this work, we augment the conversational strategies and retraining paradigms of that agent, describe the agent in detail, and conduct human studies on Mechanical Turk with our improved conversation and retraining methods.

3. Conversational Agent

We present a conversational agent that improves parsing and perception through task-oriented dialog (Figure 3).[2] We describe algorithms inspired by our past work (Thomason et al., 2015, 2016, 2017) to conduct conversations that both clarify missing parts of a user's written command and gather predicate-object labels for physical objects in the agent's environment (Thomason et al., 2019). These conversations are used to induce soft-aligned training data between user utterances and grounded *denotations*, from which we extrapolate pairings of user utterances with underlying, abstract semantic forms. These latter pairs enable us to further train the agent's semantic parser (Figure 3, *Q1*) and perceptual concept models (Figure 3, *Q2*). We restrict the agent's understanding to a fixed set of actions and their semantic arguments (A.1), and we assume the initial resources provided to the agent are insufficient to represent the entire domain (i.e., there are words to learn) (A.2).

Table 1 gives the recurring terminology and symbols used in this article. The following subsections describe the agent's semantic parser, perceptual concept models, language grounding formalism, agent dialog policy, and learning paradigm.

3.1 Semantic Parser

Semantic parsing translates human language commands to meaning representations that a robot can reason with. Learning to perform these translations often involves an ontology of concepts, a lexicon that maps words to compositions of those concepts, and examples

2. The source code for this conversational dialog agent, as well as the experiments described in the following section, can be found at `https://github.com/thomason-jesse/grounded_dialog_agent`

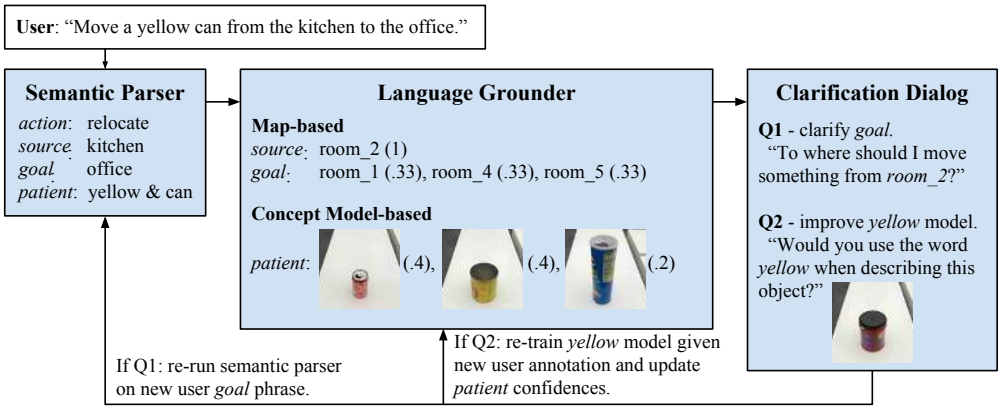
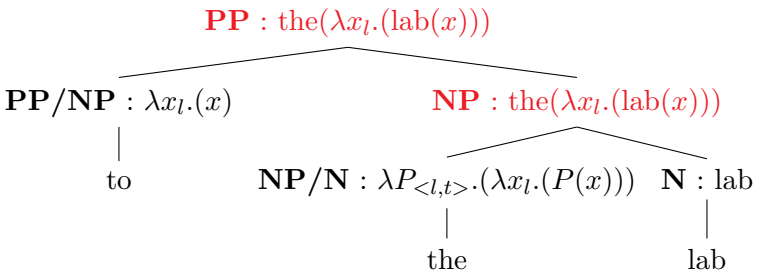


Figure 3: Conversational agent. User commands are parsed into semantic roles (**left**), which are grounded (**center**) using either a known map (for rooms and people) or learned concept models (for objects) to a distribution over possible satisfying referents (e.g., all rooms that can be described as an "office"). A clarification dialog (**right**) recovers from ambiguous or misunderstood roles (e.g., *Q1*) and to improve concept models on the fly (e.g., *Q2*).

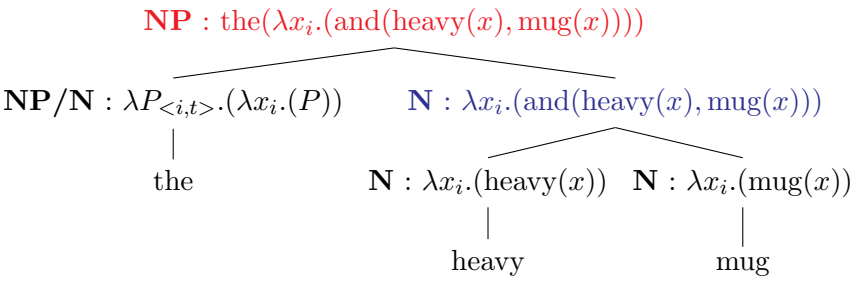
Symbol	Description
$\mathcal{DA}$	Dialog agent
CCG	Set of Combinatory Categorial Grammar syntax categories
D	Set of pairs of language strings and semantic forms
$\mathcal{C}$	Set of sensorimotor contexts of explored objects
$\mathcal{D}$	Grounding procedure producing denotations
$\mathcal{G}$	Perceptual classifier
$\mathcal{L}$	Lexicon
P	Set of predicates
$\mathcal{P}$	Parser
$\mathcal{O}$	Ontology
S	Set of semantic forms
$\bar{S}$	Parser score vector
T	Set of word tokens
$\mathcal{U}$	Top-down parsing generation procedure
$\mathcal{X}$	Explored object feature vector
θ	Parser parameter vector
κ	Confidence of a perceptual classifier
π	Dialog policy
ϕ	Parser feature vector

Table 1: Symbols used throughout this article.

of sentences paired with their meaning representations. These paired examples are used to learn a set of parameters for performing structured inference to translate the linear sequence of words into a compositional meaning structure.



(a) Function application used twice to compose the meaning of *to the lab*.



(b) Function application and merge used to compose the meaning of *the heavy mug*.

Figure 4: CCG-driven λ-calculus parses of the expressions *to the lab* and *the heavy mug* that demonstrates function application and merge parsing rules.

In this work, we use the Combinatory Categorial Grammar (CCG) formalism (Steedman & Baldridge, 2011) and Cocke-Kasami-Younger (CKY) chart parsing (Younger, 1967). Our parser is functionally similar to the University of Washington Semantic Parsing Framework (Artzi & Zettlemoyer, 2013a), but we use a unification-based training procedure rather than a template-based one. We add ontological entries dynamically (for example, when a new perceptual concept like *red* is used for the first time). We use word embeddings to guide the parsing of out-of-vocabulary words.

Given the syntactic tree of a sentence, meaning representations can be propagated from the leaf level to the root to find the composed meaning (Figures 2 and 4).[3] Here, we briefly describe the semantic parser's design and training paradigm.

The parser $\mathcal{P} : \mathbb{P}(T) \times \mathcal{L}_{\mathcal{O}} \rightarrow S$ takes in a sequence of word tokens $x = (x_1, x_2, \ldots, x_N) \in \mathbb{P}(T)$ from the set of all sequences composed of elements in T, the set of all word tokens, and a lexicon $\mathcal{L}_{\mathcal{O}}$ for ontology $\mathcal{O}$. It produces as output a semantic parse $s \in S$, the set of all semantic parses possible from the ontology $\mathcal{O}$.

In this work, the ontology includes constants—such as people (*alice*), objects (o_3), and rooms (r_2)—and predicates—such as room types (*lab*), relations (*westof*), and perceptual concepts (*small*). Each constant is associated with a type, such as *person* or *item*. Each predicate specifies a number of expected arguments and their types and returns true or false. For example, *red* is a perceptual concept predicate that tells whether its argument, which is a constant of type *item*, is red in color. We assume the user always specifies a

3. The source code for this parsing framework is available at `https://github.com/thomason-jesse/tsp`.

Word	**CCG Category**	**Semantic Form**
bring	M/NP/NP	$\lambda y_p.(\lambda x_i.(\text{bring}(x, y)))$
bring	M/PP/NP	$\lambda x_i.(\lambda y_p.(\text{bring}(x, y)))$
go	M/PP	$\lambda x_l.(\text{walk}(x))$
move	M/PP/PP/NP	$\lambda x_i.(\lambda y_l.(\lambda z_l.(\text{move}(x, y, z))))$
to	PP/NP	$\lambda x_p.(x)$
's	NP/N\NP	$\lambda x_p.(\lambda P_{<l,t>}.(\text{the}(\lambda y_l.($ $\text{and}(P(y), \text{pos}(x, y))))))$
next	N/PP\N	$\lambda P_{<l,t>}.(\lambda x_l.(\lambda y_l.($ $\text{and}(P(y), \text{adjacent}(x, y)))))$
david	NP	david
lab	N	lab
metallic	N	metallic

Table 2: Sample lexical entries created from the utterances given in the initialization phase of the evaluation (Section 4). The multiple entries of *bring* facilitate constructions like *bring coffee to Bob* versus *bring Bob coffee*. The syntactic category M represents an *iMperative* statement. Following each λ instantiation is a type (e.g., *p*erson, *i*tem), which helps constrain the search during parsing. Expected input and output types are specified in brackets ($< input, output >$). The special symbol *the* functions to select a single atom among those specified by its λ-headed argument.

discrete action the robot can perform. Thus, the desired action is therefore representable in this semantic formalism (A.4).

The lexicon is a data structure that contains information about individual word token CCG categories and relations to the ontology. The lexicon is a set of triples, $(t, c, s) \in \mathcal{L}_{\mathcal{O}}$, where $t \in T$ is a word token, $c \in CGG$ is a CCG syntax category, and $s \in S$ is a meaning representation composed of predicates, constants, and λ abstractions. Table 2 lists example entries from the initialization phase of the evaluation (Section 4).

The meaning of a sequence of tokens x is the root of its semantic parse tree. The CCG categories of x determine how the semantic forms of the tokens compose to form a tree. Categories combine by *function application* (Figure 4a) and *merge* (Figure 4b).

The examples in Figure 4 show some bare nouns and adjectives (with syntactic category N) both with and without λ-instantiated variables inside. For every bare noun entry in the lexicon, we apply a *type-raising* operation to create two more entries: one with a λ variable of the expected type as an argument, and another nested as a child of a *the* predicate with said λ child (e.g., $\text{the}(\lambda x : i.(\text{lab}(x))))$. Although type-raising increases the size of the lexicon, it enables the parsing of ungrammatical phrases such as *bring cup*.

Given a set of pairs $(x, r) \in D$ of token sequences and root semantic forms, we train a statistical parser (Liang & Potts, 2015). We define a feature vector $\phi(x, y) \in \mathbb{R}^N$. Entries i in $\phi(x, y)_i$ represent relationships between the input utterance x to the latent semantic tree y (rooted at the corresponding r), or between constituent nodes of the tree y. Table 3 shows a summary of these features. Next, we discuss how the latent tree y is inferred from the pair (x, r) to facilitate training.

Feature	Description
CCG given token	Entries counting each assignment of a CCG category to an input token
CCG rules	Entries counting each production rule $c_k \rightarrow c_i, c_j$ in the syntax tree of y
Sem given token	Entries counting each assignment of a lexical entry to a polysemous input token
Shallow Sem	Entries counting parent-child relationships in the semantic root of y
Skip given token	Entries indicating whether each token was skipped

Table 3: The features $\phi(x, y)$ we extract from a sequence of words x paired with latent tree y; each feature appears as a count in $\mathbb{N}^0$ of the times it appears in $\phi(x, y)$. *Tokens* are words in the input utterance x. *CCG* categories are syntax categories. *Sem* are semantic forms in the lexicon, independent of their CCG categories. *Shallow Sem* are parent-child relationships in semantic forms that include argument position.

3.1.1 Parsing an Input Utterance

We score parse trees $\hat{y}$ as the dot product $\theta \cdot \phi(x, \hat{y})$. Each subset of feature types is parameterized by a probability distribution, such that the likelihood of the parse $\hat{y}$ given x is $\theta \cdot \phi(x, \hat{y})$. To avoid numerical underflow, our computations are done in log-likelihood space. For an input token sequence x, we execute $P(x)$ as a beam search over possible parses of the token sequence x and find the $\hat{y}$ parse with the minimum negative log-likelihood.

Parsing proceeds in stages, with each stage greedily scoring pieces of a candidate parse by examining relevant subsets of θ, then passing the current best on to the next stage. The depth of candidates is limited to a fixed-size beam. These stages are: picking skipwords in x to create a subsequence for parsing, creating a CCG syntax tree C_x, choosing semantic leaves for each x, and composing leaves through C_x to form a completed semantic tree $\hat{y}$.[4]

3.1.2 Training the Semantic Parser

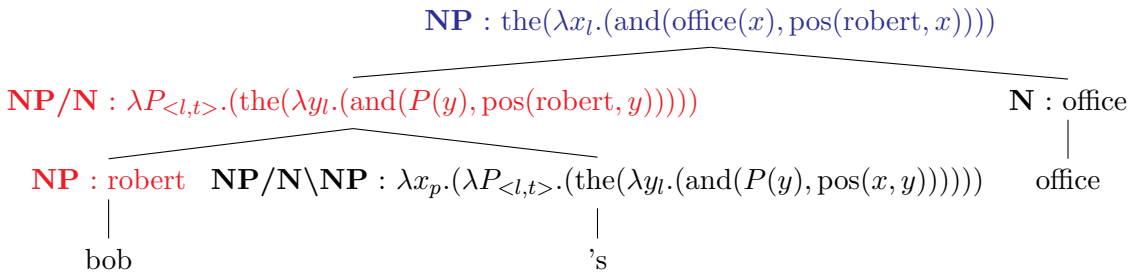
To learn values for the parameter vector θ, we consider all the pairs $(x, r) \in D$, and we run the parsing beam search outlined above until both the maximally scoring parse y^* and the correct parse $\hat{y}$ (rooted at r) are returned:

$$y^* = \text{argmax}_{\hat{y} \in P(x)}(\theta \cdot \phi(x, \hat{y}));$$
$$y \in P(x) | root(y) = r.$$

That is, y^* is the latent parse tree produced by $P(x)$ that maximizes the current vector of parameter weights, θ, while y is the latent parse tree produced by $P(x)$ rooted at the target semantic meaning r. If $y^* = y$, then θ correctly guides $\mathcal{P}$ to produce the correct parse. A correct latent parse y is not always found in the beam search approximation of $P(x)$.

If y is not found, $\mathcal{P}$ may not have the lexical entries necessary to translate x to r. We therefore produce new lexical entries to bridge the gap. Given a partial parse, $Y =$

4. See the source code for details of the implementation and hyper-parameters for this process.



(a) A top-down search from the known root on the noun phrase *bob's office* reveals that *bob* is a nickname for the person *robert*.

Word	CCG Category	Semantic Form
bob	NP	robert

(b) A new lexical entry induced from the *top-down* completion of the parse of *bob's office*.

Figure 5: An example of *top-down* completion for the parse of *bob's office* and the resulting inferred lexical entry.

$(\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_k)$, a sequence of parse trees rooted at spans of x, and the target root r, we can perform a *top-down* search from r by reversing the CCG function application and merge rules (Figure 5). This top-down parsing is conceptually similar to previous work on inducing CCG grammars using higher-order unification operations (Kwiatkowski, Zettlemoyer, Goldwater, & Steedman, 2010). We denote the lexical entries derived from this procedure as $\mathcal{U}(Y, r)$. We add $\mathcal{U}(Y, r)$ to $\mathcal{L}$, and the resulting latent tree y is derived.

If $y \neq y^*$, we update θ as follows. The parser $\mathcal{P}$ maintains a running score vector, $\bar{S}$, of every feature. After $\bar{S}$ is initialized (e.g., to all zeros), it is updated during parser training using stochastic gradient descent. In particular, given $\phi(x, y^*)$ and $\phi(x, y)$, $\bar{S}$ is updated as:

$$\bar{S}_i \leftarrow \bar{S}_i + \alpha(\phi(x, y) - \phi(x, y^*)_i), \tag{3}$$

for a learning rate α. This is similar to the hinge-loss used in previous work (Liang & Potts, 2015). As suggested in that work, we set $\alpha = \sqrt{|D|}$. Given this vector of scores, we calculate θ as a collection of negative log probabilities over feature types (Table 3).

This training procedure iterates over the examples in D in random order, updates $\bar{S}$ for each pair in D, and updates θ after passing over all of D. Training is repeated for a fixed number of epochs.

3.1.3 Using Word Embeddings to Aid Semantic Parsing

At test time, a known root r is unavailable to perform top-down parsing. We use word embeddings (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) to augment the lexicon at test time to recover from out-of-vocabulary words. This idea is similar in spirit to previous work (Bastianelli, Croce, Vanzo, Basili, & Nardi, 2016), but it is formally integrated into our parsing pipeline (Thomason et al., 2019).

The lexicon is augmented with induced entries, visible only during the parsing of x, mapping x_i to every CCG category and semantic form pair in the lexicon. Word embedding

Figure 6: **Left:** The robot platform and arm used in our demonstration and for exploring objects. **Right:** The objects explored by the robot for grounding perceptual predicates.

distance between x_i and the tokens defined in L is used to parameterize these entries. Values for θ corresponding to x_i are weighted by the cosine similarity between x_i and every word $t \in T$ in the lexicon $\mathcal{L}$, $\text{sim}(x_i, t)$. A similarity-weighted penalty is added to the θ score. After this process is repeated for every unseen x_i in x, parsing proceeds, with x_i able to take on any known meaning. This augmentation helps recover from out-of-vocabulary words, such as *grab* being used for *deliver*.

3.2 Multi-modal Perceptual Representations

Once a command has been translated into a semantic form, it needs to be *grounded* to actions, objects, and rooms in the real world. Perceptual concepts like *red* and *heavy* require considering sensory perception of physical objects. A robot arm performs a set of exploratory behaviors on each object to extract physical features across various sensorimotor contexts. Then, multi-modal concept models are learned to connect these features to words.

We used a Kinova MICO arm mounted on a custom mobile base, pictured in Figure 6 (Left). Perception includes joint effort sensors in each of the robot arm's motors, a microphone mounted on the mobile base, and an ASUS Xtion Pro RGBD camera. The set of objects O used in our experiments consists of 32 common household items, shown in Figure 6 (Right). Some items contain liquids or other contents (e.g., coffee beans), while others are empty.

The robot explores the objects using the methodology described in prior work (Sinapov, Schenck, Staley, Sukhoy, & Stoytchev, 2014a; Sinapov et al., 2016). Briefly, the robot arm takes 7 distinct actions per object: *grasp*, *lift*, *hold*, *lower*, *drop*, *push*, and *press*, shown in Figure 7. While executing each action, the robot records the sensory perceptions from *haptic* (i.e., joint efforts) and *auditory* sensory modalities. During the *grasp* action, the robot also records *proprioception* in the form of its finger positions. Joint efforts and joint positions are recorded for all 6 joints at 15 Hz. The auditory sensory modality is represented as the Discrete Fourier Transform. The continuous haptic and proprioceptive data are binned by frequency to achieve the dimensionalities shown in Table 4. The robot also performs the *look* action to produce three feature representations: 1) an RGB color histogram of the object using 4 bins per channel; 2) fast point feature histogram (*fpfh*) shape features (Rusu, Blodow, & Beetz, 2009), as implemented in the Point Cloud Library (Aldoma et al., 2012);

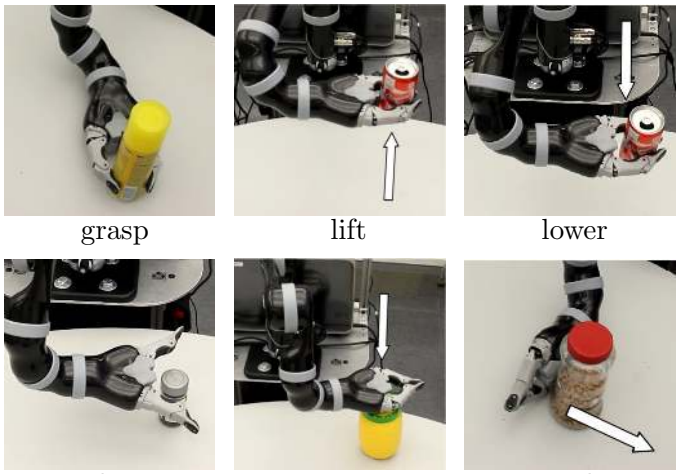


Figure 7: Exploration behaviors the robot used. The *hold* behavior (not shown) was performed after the *lift* behavior by holding the object in place for half a second.

Behavior	**Modality**		
	color	**fpfh**	**vgg**
look	64	308	4096
	audio	**haptics**	**proprioception**
grasp	100	60	20
{drop, hold, lift, lower, press, push}	100	60	

Table 4: The number of features extracted from each *sensorimotor context*.

and 3) deep visual features from both the penultimate and prediction layers of the ResNet-152 network (He et al., 2016a). The first two types of features are computed using the segmented point cloud, while the deep features are computed using the 2D image.

Each robot action produces two or three different sensory signals. Each combination of an action and a sensory modality is a unique sensorimotor context. In our experiment, the set of contexts $\mathcal{C}$ is of size $2 \times 3 + 6 \times 2 = 18$. The robot performed its full sequence of exploratory actions on each object 5 different times, with the object rotated to a new starting angle each time. Given a context $c \in \mathcal{C}$ and an object $o \in \mathcal{O}$, we denote as $\mathcal{X}_o^c$ all feature vectors observed with object o in context c. We connect these feature representations of objects to language labels by learning discriminative classifiers for each concept.

The predicates P we consider are based on either static facts or perceptual concepts. We partition P into nonintersecting subsets P_s, the subset of predicates referring to static facts, and P_c: the subset of predicates referring to perceptual concepts. Given a concept predicate $p \in P_c$ and objects labeled as positive or negative for p, we train an ensemble

of SVM classifiers, one per sensorimotor context $c \in \mathcal{C}$. We call this ensemble a *predicate classifier*, $\mathcal{G}_p$. Each context c decision on object o is the majority vote of SVM decisions on observations $\mathcal{X}_o^c$. The decision $\mathcal{G}_p(o) \in \{-1, 1\}$ for an unlabeled object $o \in O$ for p is obtained by weighted majority voting of these contexts, with a weight for each context equal to its reliability, which is estimated using leave-one-object-out cross validation on the available labeled objects for p. The decision is accompanied by a confidence $\kappa_p(o) \in [0, 1]$ that is equal to the weighted sum of these reliabilities with weights set to the Cohen's kappa agreement between the true and predicted labels during cross-validation. Negative κ values are rounded to zero.

If multiple labels are given for the same predicate and object, the majority class label is used during training. Because the robot interacts with multiple people, differing opinions occur (e.g., people differ on whether color words like *red* and *orange* apply). In the event of a tie, no label is assigned. We remove perceptual feedback from uncooperative users (Section 4.1). To avoid premature overfitting, initial label votes of positive and negative are added to every predicate/object combination (effectively Laplace-1 smoothing the labels since we expect some noise in human feedback), though some noisy concept models remain (Table 14). The confidence κ values also drive an opportunistic active learning strategy for improving concept models during conversations (Section 3.4).

3.3 Language Grounding

With learned perceptual concept models, the agent can ground semantic forms to execute given commands. Given a semantic root $r \in S$, grounding instantiates all λ expressions in r to predicates and constants in the ontology $\mathcal{O}$, resulting in a set of *denotations* $g_i \in S$ and corresponding confidence scores $c_i \in [0, 1]$. The set of denotations is a strict subset of the set of semantic forms S containing no λ expressions. We assume this set contains all possible referents (i.e., not a previously unknown place, object, or person in the ontology) (A.5). We define a grounding procedure as $\mathcal{D} : S \times P \rightarrow S \times [0, 1]$, such that $\mathcal{D}(r) = \{(g_1, c_1), (g_2, c_2), \ldots, (g_k, c_k)\}$, where r is any semantic form and g_i are the corresponding semantic forms with their λ variables instantiated and associated with confidence c_i based on the predicates invoked from P. A confidence score of zero means the grounding is not viable, i.e., there is no way to satisfy the constraints on the λ variables, while a confidence score of 1 means the grounding is unambiguous, i.e., it is the only solution to instantiating the λ variables.

We implement a recursive grounding procedure. For every λ variable encountered in the form r, recursive calls ground the child of the λ-headed parse with every possible value for that λ variable filled in (drawn from the λ variable's type). The base condition for this recursive procedure is a predicate $p \in P$ applied to some constant $a \in \mathcal{O}$. For $p \in P_s$, the set of static fact predicates, $p(a)$ is retrieved from a table as $g = \textit{true}$ or $g = \textit{false}$ with confidence $c = 1$. Static facts include room types (*office*) and relations (*owns(robert, room1)*). For $p \in P_c$ perceptual predicates, $p(a)$ is evaluated by a concept model for a, an object instance in the real world. Two groundings are returned: $g_j = \textit{true}$ with $c_j = \kappa_p(a)$ when $p(a) = 1$, and $g_k =$*false* with $c_k = (1 - \kappa_p(a))$ when $p(a) = -1$. For the *and* predicate, *true* is returned if its children match, with confidence equal to the product of the child confidences. The product of confidences can be used because we assume independence

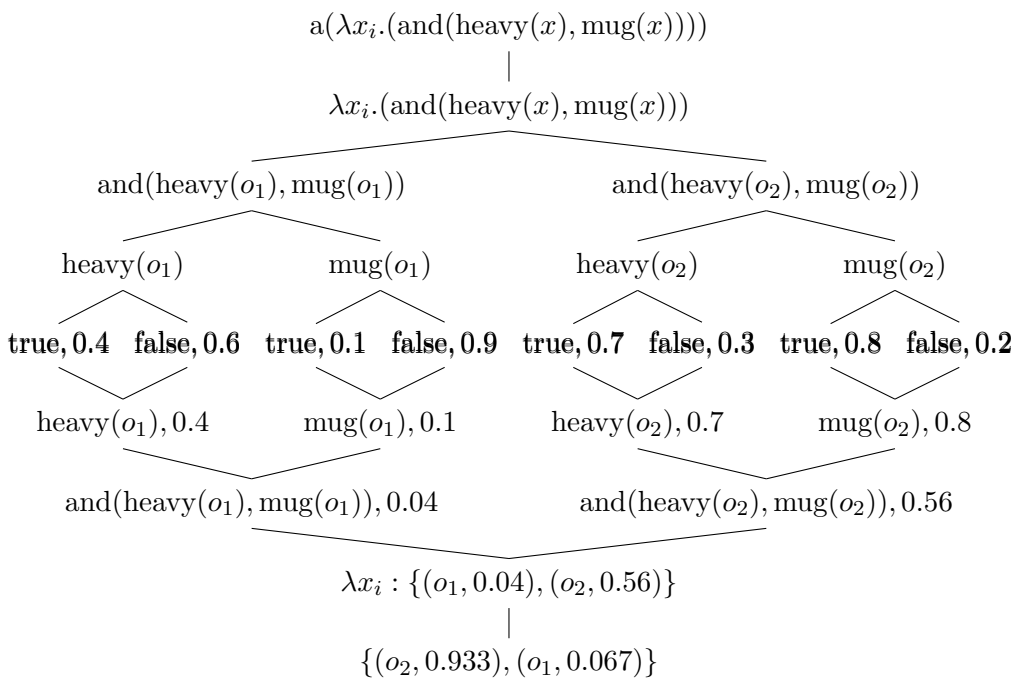


Figure 8: Grounding procedure for the semantic parse of *a heavy mug* for a world with only two objects, o_1 and o_2. The λ variable is instantiated for every object, and the confidence of the predicates applied to those instances is propagated to the root. The *a* predicate induces a probability distribution over possible denotations of the λ variable.

between concepts A.6. We normalize these confidences to form a probability distribution over possible instantiations.

Figure 8 shows an example denotation of *a heavy mug* for a subset of two objects. The possible denotations are used as part of an update procedure for helping the agent infer the user's intent, as described in the next section.

3.4 Dialog Policy

The proposed dialog agent $\mathcal{A}$ extends our previous work (Thomason et al., 2019) with a dialog recovery strategy. The agent engages in dialogs to refine its understanding of natural language commands by inducing additional training data for its parsing and perceptual grounding components. In this section, we discuss the dialog strategies employed.

3.4.1 Clarification Dialog Policy

A human user commands the agent to perform a task. The agent maintains a belief state that models the unobserved true task to be inferred. Parsing and grounding of each user utterance x results in a set of denotations and confidence values (g_i, c_i). We update the agent's belief state from these pairs and engage in a clarification dialog to resolve uncertainty.

The agent's belief state, $\mathcal{B}$, is a mapping from semantic roles (components of the task) to probability distributions over denotations. The belief state models the *action*, *patient*,

B **max per role** (***action***, ***patient***, ***recipient***, ***source***, ***goal***)	**Min Prob** B **Role**	**Question**	**Type**
$(\varnothing, \varnothing, \varnothing, \varnothing, \varnothing)$	All	What should I do?	Clarification
$(\text{navigate}, \varnothing, \varnothing, \varnothing, r_1)$	*action*	You want me to go somewhere?	Confirmation
$(\text{deliver}, \varnothing, p_1, \varnothing, \varnothing)$	*patient*	What should I deliver to p_1?	Clarification
$(\text{relocate}, \varnothing, \varnothing, \varnothing, \varnothing)$	*source*	Where should I move something from on its way somewhere else?	Clarification
$(\text{relocate}, o_1, \varnothing, r_1, r_2)$	-	You want me to move o_1 from r_1 to r_2?	Confirmation

Table 5: Samples of the agent's static dialog policy π for mapping belief states to questions.

recipient, *source*, and *goal* semantic roles. Different actions utilize different roles. For example, the *navigate* action takes only a *goal* room, while the *deliver* action takes both a *patient* object and a *recipient* person.

In our experiments, the possible actions are *navigate* (move to a different room), *deliver* (take an object to a person), and *relocate* (move an object from one room to another). The belief for the *action* role is initialized uniformly between these actions. The other role beliefs are initialized with half of the initial probability mass on an *unknown* constant, $\varnothing$, indicating that the role is not known or is not necessary for the action. The remaining half of the probability mass is uniform across all constants that can fill the role.

The belief state is updated on a per-role basis. For each role, a probability distribution is induced from the set of denotations $\mathcal{D}(\mathcal{P}(x))$. For example, from the command processed in Figure 3, the distribution over *action* has all mass on *relocate*, while the distribution over *patient* distributes mass across three potential objects. We denote these distributions $\mathcal{B}_x$, and we update the agent's belief as:

$$\mathcal{B}(r, a) \leftarrow (1 - \rho)\mathcal{B}(r, a) + \rho\mathcal{B}_x(r, a), \tag{4}$$

for every semantic role r and every constant a. The parameter ρ controls how much to weigh the new information versus the current belief (in our experiments, we set $\rho = 0.5$).

The dialog agent poses questions to the user regarding different semantic roles (e.g., Figure 3 *Q1*). The highest-probability constant for every semantic role in the current belief state $\mathcal{B}$, and an indicator of which of these has the least probability, are input to a static dialog policy, π. The policy chooses a follow-up *confirmation* or *clarification* question. Confirmations ask whether a certain constant in a role is correct. Clarifications ask the user to rephrase a referring expression. Table 5 shows some examples of the policy π.

After confirmation questions, the confirmed $\mathcal{B}_x$ constant(s) receive the whole probability mass for their roles, and ρ is set to 1 for the update in Equation 4. If a user denies a confirmation question, $\mathcal{B}_x$ is constructed with the constants in the denied question given zero probability weight for their roles and other constants given a uniform weight. However, a previously confirmed role never loses probability mass (for example, once a user confirms *navigating* as the action, if the agent asks whether room r_1 is the goal and the user says no,

only r_1 loses mass). A conversation concludes when the user has confirmed every semantic role, and these roles parameterize the action to be taken (A.1).

3.4.2 Recovery Strategy

We use a recovery strategy for dialogs that are becoming repetitive by imposing restrictions on question asking. The agent cannot ask two confirmation questions in a row about the same semantic role, e.g., *You want me to go to room 1?* followed by *You want me to go to room 2?* Additionally, the agent can ask for an open-ended clarification of a role, e.g., *Where should I go?*, at most two times. These rules prevent the agent from exhaustively enumerating the possible values of a semantic role.

To recover when a role is unconfirmed and the agent is out of question options, we enable the agent to ask the user to select the answer by scrolling through a list of potential answers. We call these *enumeration* questions because a scrollable list is enumerated to the user in a non-language interface. The agent asks, for example, *Where should I go?*, and offers an image of a floor plan with one room marked with an X, with left and right scroll buttons to choose different room referents from all possible. These questions make it faster for the agent to enumerate possibilities than as a series of confirmation questions, and they remove the risk of continued misunderstanding by repeating open-ended clarification questions.

Enumeration questions are possible with a simulated dialog agent as well as on a physical robot platform given a touchscreen or other non-verbal interaction mechanism, making it plausible as an option moving forward in these types of embodied systems. The recovery strategy puts an upper bound on the number of questions in a dialog and increases the amount of training data that can be induced from conversations by offering users a non-language-based method to select the referent of their previous language utterances.

3.4.3 Detecting Perceptual Words and Synonyms

Humans can use words an agent has never heard before. Some of these are perceptual concepts—words that need to be grounded in the physical world. The agent hypothesizes whether each new word might be perceptual based on known perceptual words, then asks for confirmation.

For an input user utterance x, the agent identifies all words x_i for which the parser, $\mathcal{P}$, does not yet have lexical entries. If one of the nearest neighbors (in our work, the agent takes the nearest 3) of x_i in $\mathcal{P}$ according to word-embedding distance has a semantic form involving a perceptual predicate $p \in P_c$, the agent asks the user whether the unseen word x_i is also perceptual in nature. The question posed is: *I haven't heard the word 'x_i' before. Does it refer to properties of things, like a color, shape, or weight?*. If the user answers *no*, x is passed on to $\mathcal{P}$, where x_i may be resolved in other ways (such as using induced lexical entries; Section 3.1). If the user answers *yes*, the agent attempts to discover whether x_i is a synonym of an already known perceptual concept or a novel one.

The agent ranks the nearest neighbors of x_i by distance and sequentially asks the user whether the next nearest neighbor t_p is a synonym of x_i. Specifically, the question posed is *Does 'x_i' mean the same thing as 't_p'?*. If an affirmative answer is given for any neighbor, t_p, new lexical entries are created in L for x_i matching those for t_p, that share an underlying

ROBOT	What should I deliver to this person?
YOU	a tall metal can
ROBOT	I haven't heard the word 'tall' before. Does it refer to properties of things, like a color, shape, or weight?
YOU	yes
ROBOT	Does 'tall' mean the same thing as 'tower'?
YOU	no
ROBOT	Does 'tall' mean the same thing as 'white'?
YOU	no
ROBOT	Does 'tall' mean the same thing as 'long'?
YOU	yes

Figure 9: Web interface example of learning that the word *tall* is a synonym of *long* in this domain when describing objects.

perceptual concept model. For example, in our experiments, the previously unseen word *tall* was added as a synonym for the known word *long* (Figure 9).

If no synonym is identified, a new ontological concept is created to represent x_i. In particular, a new predicate for x_i, p_{x_i} is added to $\mathcal{O}$ (e.g., $p_{x_i} \in P_c$), and corresponding lexical entries mapping x_i to p_{x_i} are added to $\mathcal{L}$. For example, in our experiments, the color concept word *red* was added with a new ontological predicate to represent it.

After these changes to $\mathcal{O}$ and $\mathcal{L}$, the agent resumes parsing. We both expand the ontology to represent new concepts and partially account for concept synonymy, ameliorating the concept-object label sparsity introduced by treating all new words as new concepts (A.6).

3.4.4 Opportunistic Active Learning during Conversation

Concept models like *heavy* improve as more objects are labeled with them. Building on our prior work (Padmakumar, Stone, & Mooney, 2018; Thomason et al., 2017), we introduce opportunistic active learning questions (e.g., Figure 3 *Q2*) as a sub-dialog routine for the agent. The agent queries about objects *local* to the human and the robot (e.g., objects in the room where the conversation is happening) to refine its perceptual concept models before applying them to *remote* test objects (e.g., those in a different room). We call local objects the *active training set* and remote objects the *active test set*.

Given command x, a sub-dialog for improving perceptual concept models begins (Figure 10). This sub-dialog starts with the agent saying, *I'm still learning the meanings of some words. I'm going to ask you a few questions about these nearby objects before we continue.* We partition P_c, the set of all perceptual concept predicates, into P_C^x, the predicates present in the command, and $\bar{P}_C^x$, those that are not. A predicate p is considered "in the command" if it is present in the logical form of any token $x_i \in x$. To determine potential active-learning queries, we use the confidence $\kappa(p, o)$ for $p \in P_C$ and $o \in O$, a form of least-confident value uncertainty sampling (Culotta & McCallum, 2005). When there are too few labeled objects in the positive and negative classes to perform cross-validation, we set $\kappa(p, o) = 0$ and the corresponding decision to $d(p, o) = -1$.

We set a maximum number of questions, m, to ask (in our experiments, we set $m = 3$). Questions are first asked about predicates in the command $\hat{P}_C = P_C^x$. A question cannot be

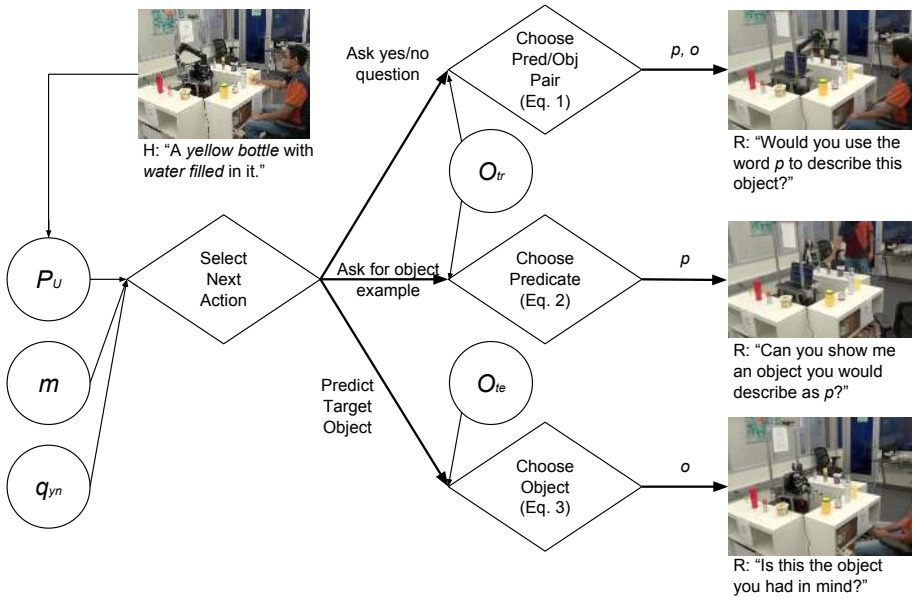


Figure 10: Sub-dialog for improving perceptual concept models using questions selected with an opportunistic active learning strategy. Here, $\hat{P}_C$ are the predicates the dialog agent can query, m is the maximum number of sub-dialog questions, p^* is a predicate selected to query, and o^* is an object selected to query a label with respect to p^*.

formulated for a predicate if all objects in O_{tr} are already labeled for that predicate. The agent queries with $\hat{P}_C = \bar{P}_C^x$ if O_{tr} are labeled for all P_C^x.

The agent calculates the average confidence $K_{te}(p)$ of each $p \in P_C^x$ on the *active test set* of objects, O_{te}:

$$K_{te}(p) = |O_{te}|^{-1} \sum_{o \in O_{te}} \kappa(p, o). \tag{5}$$

The agent decides which predicate to query next with probability:

$$pr(p) = \frac{1 - K_{te}(p)}{\sum_{q \in \hat{P}_C} (1 - K_{te}(q))}. \tag{6}$$

After selecting a predicate p^* to query by sampling from this probability distribution, the agent decides to ask for the label of a particular object or for an example.

The agent calculates the confidence of p^* among active training set objects $K_{tr}(p^*)$ similar to Equation 5. If $K_{tr}(p^*)$ falls below a fixed threshold (we use 0.7), the classifier is considered untrustworthy for determining a useful specific object to query, and the agent instead asks for a new positive or negative example. When asking for an example, if the labels for p^* on O_{tr} are majority-class positive, the agent asks for a negative example: *Among these nearby objects, could you show me one you could not use the word 'p^*' when describing, or shake your head if you could use 'p^*' when describing all of them?* Otherwise, the agent asks for a positive example: *Among these nearby objects, could you show me one you would use the word 'p^*' when describing, or shake your head if there are none?*

In the Mechanical Turk interface, users are shown images of all objects O_{tr}, and they click to answer. The *head shaking* behavior refers to a button on this interface (*shake head*) that labels all objects O_{tr} as positive or negative examples, respectively. When embodied on the physical robot, the agent recognizes the user touching objects to select them or verbally saying *all* or *none*, respectively (with questions modified to match this expectation).

If $K_{tr}(p^*)$ exceeds the confidence threshold, the trained p^* classifier is used to identify the object $o^* \in O_{tr}$ whose label is currently least certain:

$$o^* = \text{argmin}_{o \in O_{tr}}(\kappa(p^*, o)). \tag{7}$$

This question is presented as, *Would you use the word 'p^*' when describing o^*?* In the Mechanical Turk interface, o^* in the text is replaced with *this object*, and an image of the object is shown. When the agent is embodied in a robot, o^* is described as *this object*, and the robot points to the physical object on a nearby table. If no o^* can be identified (because all training objects are labeled for p^* already), the agent samples the p^* to query from $\hat{P}$.

Using this sub-dialog, the agent can query the user for labels on the objects O_{tr} to improve its perceptual classifiers before continuing its clarification conversation.

3.5 Learning from Conversations

We extract training data for the semantic parser in the form of utterance-denotation pairs from completed conversations. These pairs represent language expressions and the world grounding they represent (e.g., *the red cup* and the physical referent object).

3.5.1 Inducing Utterance-Denotation Pairs from Conversations

Every time the agent asks the user a clarification question, the question is associated with the least-confident role r being queried (or *All* roles, if the conversation has just started). There is an alignment between the constant that should fill role r and the human's natural-language answer to the question. This alignment assumes the user is cooperative and truthful (A.3). When the conversation concludes, the final chosen task b encodes the action and arguments in its semantic roles. Thus, for every r in chosen task b, where $b(r)$ is the argument chosen for role r, we induce training pairs $(x_i, b(r))$ based on the conversation history, where x_i is a human answer to a clarification question regarding role r.

For example, suppose the agent asks, *What should I find to deliver?*, and the user answers *a heavy mug*. For the final inferred task deliver(o_1, p_1), we create a pair (*a heavy mug*, o_1), where o_1 is the denotation object referred to by *the heavy mug*. We can pair the original user command (e.g. *bring the mug to bob*) with the final task denotation, deliver(o_1, p_1). Figure 11 shows a conversation where we pair the final task denotation with the original user command and a clarification question response with a denotation role (the *goal*). For each pair, we need to infer the latent semantic form that connects them and can be used as training data for the parser.

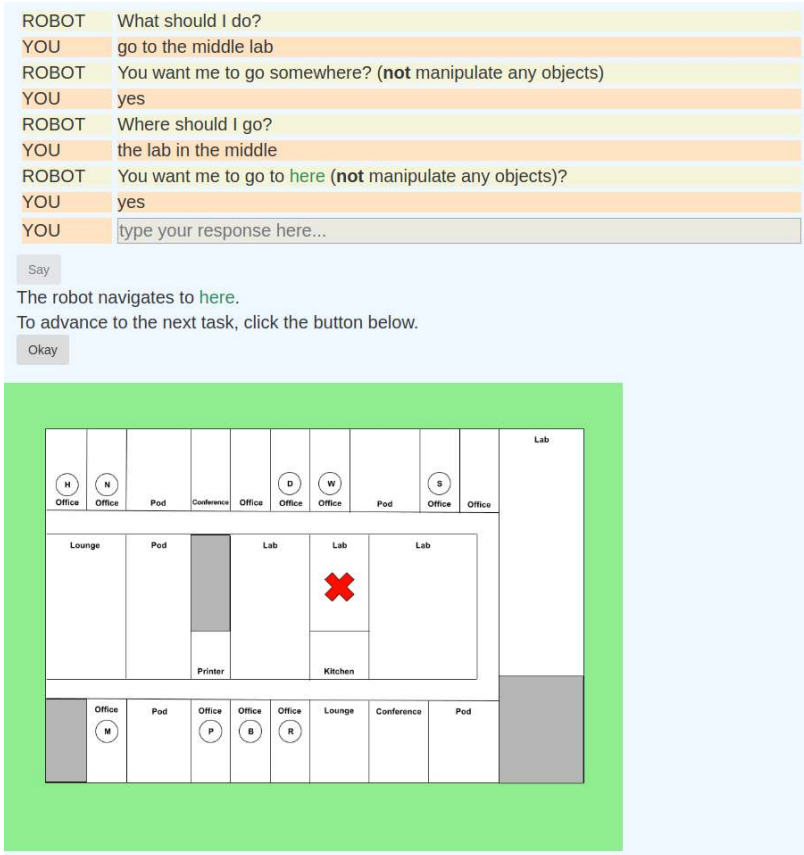


Figure 11: A completed conversation for the *navigation* task. The user first specifies the whole task and then clarifies the *goal* semantic role. The utterance-denotation pairs are thus (*go to the middle lab*, *navigate(lab2)*) and (*the lab in the middle*, *lab2*) for the ontological constant *lab2*, which represents the center lab in the diagram.

3.5.2 Inducing Utterance-Semantic Form Pairs from Utterance-Denotation Pairs

We need to induce a latent form $y \in S$ given a pair (x, g) such that $y \in \mathcal{P}(x)$ and $g \in \mathcal{D}(y)$. In other words, we need to find a semantic parse that can be derived from the input utterance and has a denotation matching the known one for that utterance.

Given (x, g) for x, a sequence of tokens, and $g \in S$, a denotation, we discover latent semantic parse $y \in S$ as follows. We produce the set of parses for x, $(y_i, c_{p,i}) \in \mathcal{P}(x)$. For every parse y_i, we find the denotations $(g_{i,j}, c_{d,i,j}) \in \mathcal{D}(y_i)$. If $g \in \mathcal{D}(y_i)$, then y_i is a potentially correct latent form. We denote matching parses by y_i^*.

We score each y_i^* based on an interpolation of its log-likelihood $c_{p,i}$ and the confidence $c_{d,i,j}$ of its denotation $(g_{i,j} = g)$. This is computed as:

$$\text{score}(y_i^*) = c_{p,i} + \log(c_{d,i,j}). \tag{8}$$

Recall that $c_{d,i,j}$ is a probability in $[0, 1]$. This gives the score of y_i^* as a log-likelihood for every potentially correct parse. In our experiments, we beam search over at most 10 forms.

$y \in \mathcal{P}(x)$	c_p	$g \in \mathcal{D}(\mathcal{P}(x))$	c_d	score(y)
the(λx_i.(heavy(x)))	-1	o_1	0.364	($g \neq o_2$)
		o_2	0.636	-1.45
the(λx_i.(and(heavy(x), mug(x))))	-1.2	o_1	0.018	($g \neq o_2$)
		o_2	0.982	**-1.22**
the(λx_i.(mug(x)))	-1.2	o_1	0.111	($g \neq o_2$)
		o_2	0.889	-1.32
and(heavy, mug)	-1.2	and(heavy, mug)	1	($g \neq o_2$)

Table 6: Example of finding the best latent parse for x : *the heavy mug* given denotation o_2. In this example, alternative parses arise from the parser choosing to skip individual words in x. In general, alternatives can arise from, for example, prepositional phrase attachment ambiguity and lexical polysemy. In this example, the parser favors skipping the noun *mug*, but the re-ranking score after grounding finds that including it improves the parse. The highest score is marked in **bold** and paired with the best latent parse for this utterance-denotation pair.

We select the highest scoring latent parse y^*, given by

$$y^* = \text{argmax}_{y_i^*}(\text{score}(y_i^*)). \tag{9}$$

We use the pair (x, y^*) as additional parser training data. Table 6 finds the latent semantic parse between *the heavy mug* and object o_2 based on the grounding in Figure 8.

4. Experiments

We evaluated our agent with hundreds of people using the Mechanical Turk interface. The agent had access to static facts about the world as well as multi-modal object feature representations against which it trained perceptual concept models. We deployed the agent with human users, who instructed it to perform three tasks: navigation (*Go to the lounge by the kitchen*), delivery (*Bring a red can to Bob*), and relocation (*Move an empty jar from the lounge by the kitchen to Alice's office*). We discuss the potential for adding more high-level tasks and relaxing the assumptions in these experiments in Section 5.

After training the agent using data from conversations on Mechanical Turk, we transferred the learned agent to a physical robot (Thomason et al., 2019). We demonstrated this physical robot and its ability to carry out the tasks in a physical office setting. While Mechanical Turk limits users to describing objects only by their visual properties, the embodied agent, initialized from Mechanical Turk, learned non-visual concepts like *rattling*.

4.1 Experiment Design

For the Mechanical Turk evaluation, the agent simulated tasks in an office with the same floor plan as the building where the physical robot operated, but with anonymized names and titles for the people on the floor. Facts about the rooms and people were static and were represented in a fixed knowledge-base. Facts about the object items were learned by building perceptual concept models.

	Navigation	**Delivery**	**Relocation**
Initialization	3	7	520
Train	18	50	3640
Test	5	15	1040
Total	26	72	5200

Table 7: Number of unique tasks assigned to the initialization, train, and test conditions. Task cardinality is a function of the number of locations (26), people (9), and object items (8). For relocation, the source and goal locations always differ.

We fixed 8 of the 32 objects explored in prior work (Sinapov et al., 2016) as possible arguments to the tasks for our experiments (selected at random), and we used the remaining 24 as training objects available for opportunistic active learning queries for learning concept models. We randomly split the set of possible tasks into initialization (10%), train (70%), and test sets (20%) (Table 7).

Including the same 8 objects in all sets as targets ensured that the perceptual concepts needed to identify those objects were the same across conditions. The objects that appeared in these tasks were never visible to the predicate concept models during training (i.e., they were considered to be *remote* by active queries).

4.1.1 Initialization Phase

We ran a small initialization phase using the web interface described in Section 4.2. Sixteen users (graduate students from several fields) gave the agent two commands each to perform three tasks (one navigation, one delivery, and one relocation) from the initialization set.

We referenced these commands to build the initial ontology $\mathcal{O}$, which represented concepts users invoked for locations and object items. For the 26 locations on our map, we created semantic map labels like *lab* and *office* (7 in total). Additionally, we annotated binary relationships like *adjacent* and *westof* (5 in total). Global relationships like *east* and *middle* were also created (5 in total). Including the relation between the 9 people on the map and their offices (ownership) and for an unoccupied room, there were 19 predicates involving locations. We included 20 initial perceptual concepts (e.g., *yellow*) that appeared in the commands.

We also annotated an initial lexicon $\mathcal{L}$. In addition to semantic map words like *lab* and *office*, we added constructions like *between* (two adjacency relations) and *northwest* (both north and west). For perception, we added the concept words and their synonyms used in the initialization commands. We also added prepositions, determiners, and action word synonyms (e.g., *visit* for the walking action). Finally, we added single lexical entries for people in the office environment based on their first names, like *robert*. In contrast, object items and rooms in the ontology could not be directly referenced because they had no single-word lexical entries. Thus, users had to describe objects using their physical properties and rooms using their labels and spatial relations to other rooms.

We annotated the commands with their corresponding semantic parses to get initialization pairs D_0. In all, creating the initial ontology, lexicon, and training pairs took about

2 hours of annotator time. We trained an initial parser with the $|D_0| = 44$ pairs annotated from the initialization. We compared agent performance with this initial parser and with perception modules that were initially blank (but could be learned on the fly using opportunistic active learning) to agent performance after retraining (Section 4.2).

A small number of commands used negation (e.g., *the pod not next to a conference room*), which we chose not to include in our parsing model. Additionally, we did not consider elided references (e.g., *from the kitchen to the conference room across the hall [from the kitchen]*, or *the office next to bob's [office]*). Because of these decisions, and several commands that would have introduced single-use ontological concepts, we did not create pairs from all 72 initialization commands.

4.1.2 Training Procedure

We now discuss the batch training procedure used to update the agent's parsing and perception modules.[5] The initial parser, denoted $\mathcal{P}_1$, was trained with only initialization pairs D_0. A baseline agent $\mathcal{A}_1$ used parser $\mathcal{P}_1$ and concept models $P_{c,1}$ with no initial object examples (i.e., the concepts were known, but not their meanings). All further learning for the parser and perception modules arose naturally from human-agent conversations.

We divided the training procedure into three phases, each associated with 8 objects from the active training set for label queries. Between phases, the parser and perception models were retrained from the conversations the agent had up to that point. Since the parser would know more words and phrases after each retraining step, users could be less inclined to converse in simpler language when the agent asked clarifying questions. Limiting the active training set to 8 objects reduces user fatigue when asking for label examples. Each phase i was conducted by agent $\mathcal{A}_i$; then, parser $\mathcal{P}_{i+1}$ and concept predicates $P_{c,i+1}$ were trained for use by agent $\mathcal{A}_{i+1}$.

For each training phase $i \in \{1, 2, 3\}$, we created 250 Mechanical Turk Human Intelligence Tasks (HITs), each with a single task prompt from the training set. Table 9 summarizes task completion counts. Worker answers to object-predict label queries and whether words were perceptual or synonyms were also considered when retraining the perceptual concept models. We consider a word w a candidate perceptual concept if at least one worker said *yes* when asked if it were perceptual in nature and labeled at least one object as a positive example of w. If a majority of users voted *yes* (that word v was a synonym for w), we added w to the lexicon as a synonym for v. If this consensus were reached for no v, then w was added to the ontology as a new perceptual concept. Finally, object-predicate labels across conversations were used to train predicate concept models $P_{c,i+1}$. In the event of a label disagreement for a predicate and object, that label was not applied during retraining.

To retrain the parser, we induced utterance-denotation pairs from completed conversations. Training $\mathcal{P}_{i+1}$ proceeded over a fixed number of epochs E, as follows. First, utterance-semantic form pairs $D_{i,e}$ were induced from the utterance-denotation pairs in conversations from phase i using the current parser $\mathcal{P}_{i+1,e}$. Next, $\mathcal{P}_{i+1,e+1}$ was trained by passing once over the pairs $D_{i,e}$. New pairs $D_{i,e+1}$ were induced using $\mathcal{P}_{i+1,e+1}$, and we repeated (in our experiments, we set $E = 10$).

5. In general, the agent can be retrained after every conversation. We retrained in batch to facilitate our evaluation, because users interacted with the agent in parallel.

Survey Prompts
The robot understood me.
The robot frustrated me.
I would use a robot like this to help navigate a new building.
I would use a robot like this to get items for myself or others.
I would use a robot like this to move items from place to place.

Table 8: Survey prompts presented after dialogs. Users indicated on a 7-point Likert scale whether they agreed or disagreed with these statements.

To summarize, we induced the latent semantic forms from utterance-denotation pairs and retrained the parser at each epoch. As the parser improved over epochs, more accurate latent forms were found. The trained parser $\mathcal{P}_{i+1,E} = \mathcal{P}_{i+1}$ was used in the next phase of the experiment for agent $\mathcal{A}_{i+1}$.

After three phases, we tested agent $\mathcal{A}_4$ with parser $\mathcal{P}_4$ and perception models $P_{c,4}$ with tasks from the unseen test set. We also tested an ablation agent, $\mathcal{A}_4^*$, with parser $\mathcal{P}_1^*$ and perception models $P_{c,4}$. Parser $\mathcal{P}_1^*$ had an expanded lexicon and ontology that included new predicates and synonyms discovered when retraining perception models $P_{c,4}$, but its training iterated only over pairs D_0. This agent served as an ablation to assess the effects of trained perception models alone versus the effects of retraining the parsing model.

4.1.3 Performance Metrics

We measured the average total number of clarification and enumeration questions the agent asked per task. We considered these metrics only when a worker confirmed the correct task since we were interested in reducing successful conversation lengths. We also gathered user answers to survey questions (Table 8). Each question was answered on a 7-point Likert scale: *Strongly Disagree* (1), *Disagree* (2), *Slightly Disagree* (3), *Neutral* (4), *Slightly Agree* (5), *Agree* (6), *Strongly Agree* (7).

4.2 Mechanical Turk Evaluation

We conducted a large-scale evaluation of our learning agent using Amazon Mechanical Turk. Extending prior work (Thomason et al., 2019), we added a non-verbal dialog recovery strategy and induced more training data from conversations (Section 3.5).

4.2.1 Web Interface

Workers engaged in a conversation with the agent and filled out a survey. At the beginning of each conversation, workers were prompted with a new task and instructed: *Command the robot with one complete sentence to solve the problem below. The robot does not understand questions, but will ask you questions of its own. The robot will show you pictures of what it thinks you want; match those to the goal pictures below.* Workers were additionally told to *Give your commands all at once, as opposed to in individual steps.*

To avoid priming workers with referring expressions, we presented tasks by describing the target state of the world pictorially. For navigation tasks, the prompt was: *Give the*

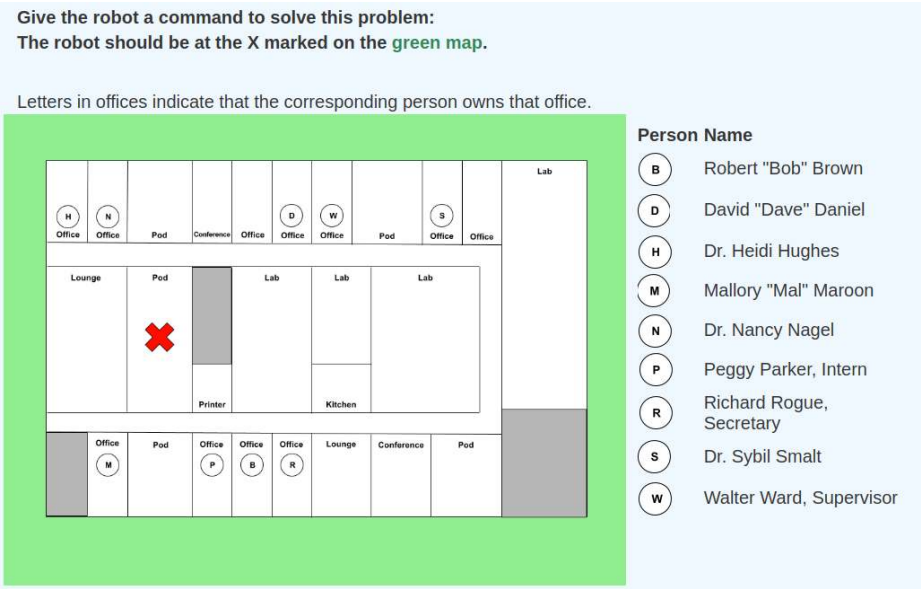


Figure 12: An example navigation task. The map on the left marks the target room. The panel on the right shows people and office ownership.

robot a command to solve this problem: The robot should be at the X marked on the green map, and the worker was presented with a green-highlighted floor map with a marker for the target room (Figure 12). For delivery tasks, the prompt was *This person needs the object shown below*; the person and object shown as pictures (right panel, Figure 12). For relocation tasks, the prompt was *The object shown below is at the X marked on the pink map. The object belongs at the X marked on the green map*; the object and corresponding color-highlighted maps.

Agent questions used color-coding reference phrases, such as *this object*, *this person*, *here*, and *there*, with a corresponding picture and matching background color. Enumeration questions offered arrows to scroll through pictorial options. Figure 13 shows a clarification and confirmation question for the *goal* semantic role of the navigation task.

The agent also asked whether perceptual concept words apply to objects. Figure 14 shows an example of such a yes/no label query. In this case, the worker typed a response. Figure 15 shows the panel used for open-ended positive/negative example queries. For these, workers clicked on one of the objects to select an answer, or on the *Shake Head* button to indicate no presented object satisfied the query.

4.2.2 Human Intelligence Task (HIT) Setup and Completion Data

Workers connecting to HITs were assigned at random to one of the three tasks (*Navigation*, *Delivery*, or *Relocation*).[6] For the train condition, we created 250 HITs per fold of active

6. Workers were paid $1 per completed HIT. Each HIT was available to submit for 2 hours after starting. The dialog-agent server enforced a time limit of 1 hour per worker. If a conversation exceeded 1 hour, the agent timed out, allowing the worker to advance to payment.

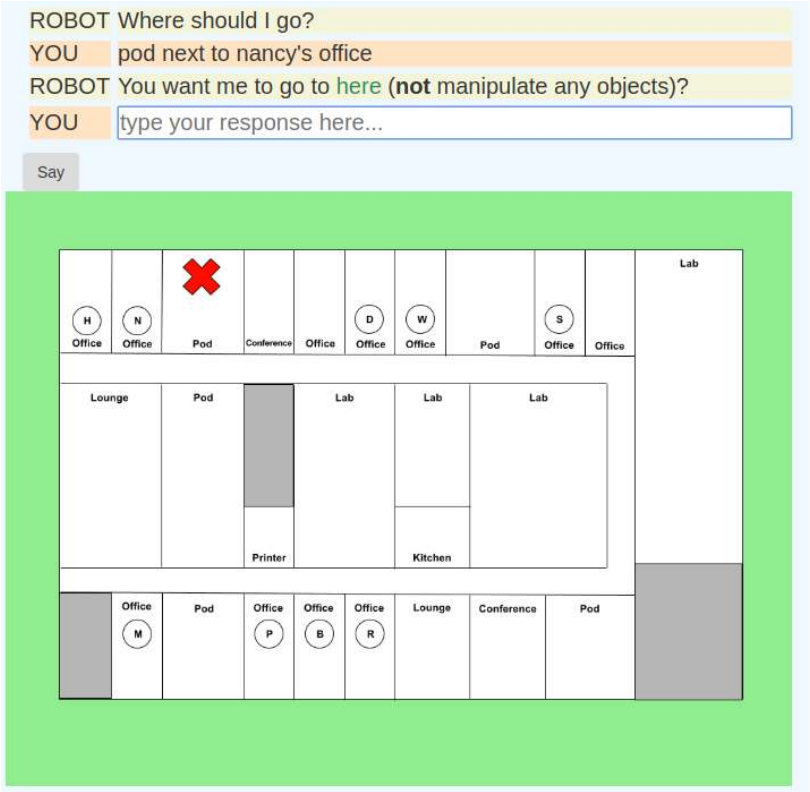


Figure 13: Example of the agent asking a confirmation question. The green word *here* refers to the green-bordered map, the understood *goal* of the command. The expression *pod next to nancy's office* can be tied to the denotation (the referent room) for parser training.

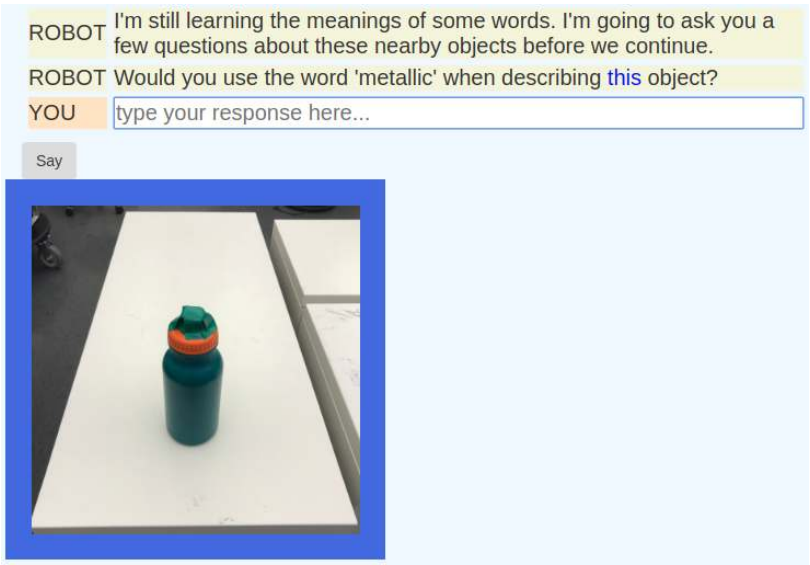


Figure 14: Web interface for the agent querying whether a predicate applies to an object. The blue word *this* refers to the blue-bordered object image.

Figure 15: Web interface for the agent asking for a positive or negative example of a predicate, for example, *Among these nearby objects, could you show me one you would use the word 'red' when describing, or shake your head if there are none?*

training set objects. The conversational dialog agent had access to word embeddings to guess the meanings of out-of-vocabulary words on the fly (Section 3.1).

For the test conditions—without parser or perception training ($\mathcal{A}_1$), with perception training only ($\mathcal{A}_4^*$), and with parser and perception training ($\mathcal{A}_4$)—we created 250 HITs per condition.[7] We enforced that each test HIT was completed by a unique worker who did not participate in the training phase. Thus, all test HITs were completed by workers who were conversing with our agent for the first time.

Table 9 shows worker counts. We used a worker's data for training if they confirmed at least one correct semantic role with the agent, for example correctly conveying a *navigation* action or correctly specifying the *goal* room. We induce training examples from partially correct commands when learning from conversations (Section 3.5) by considering only correctly specified roles (e.g., only *goal* clarification utterances), deviating from our prior work requiring an entirely correct command (Thomason et al., 2019).

The enumeration fallback questions ensure that a user can always arrive at the correct command. Thus, we considered a worker's performance data (conversation length and survey responses) for testing only if they confirm an entirely correct command. This let us focus on cases where users were cooperative (A.3).

We removed contrasting utterance-denotation pairs. If utterance U had denotation a from conversation A, and denotation b from conversation B (with $a \neq b$), the pairs (U, A) and (U, B) were both removed from the training data. This prevented intuitively

7. For the final 50 HITs of each condition, workers were deterministically assigned to the *Relocation* task, which had a lower return rate due to its length and difficulty.

Condition	Number of Workers		
	Navigation	Delivery	Relocation
	Used for Training		
Train ($\mathcal{DA}_1$)	70	72	49
Train ($\mathcal{DA}_2$)	71	70	37
Train ($\mathcal{DA}_3$)	74	66	49
	Used for Testing		
$\mathcal{DA}_1$	32	43	14
$\mathcal{DA}_4^*$	33	42	21
$\mathcal{DA}_4$	38	46	15

Table 9: The number of workers whose conversations were used for *training* and whose metrics and survey responses were considering during *testing*.

	Clarification Questions ↓					
	Navigation		Delivery		Relocation	
Agent	#Qs	%Δ (p)	#Qs	%Δ (p)	#Qs	%Δ (p)
$\mathcal{DA}_1$	7.8 ± 4.3		13.3 ± 3.2		16.5 ± 2.9	
$\mathcal{DA}_4^*$	7.5 ± 3.3	$-3.9(.76)$	**11.0** ± 3.1	$-17.1(.0015)$	**14.1** ± 3.7	$-14.5(.045)$
$\mathcal{DA}_4$	7.6 ± 2.8	$-2.6(.82)$	**10.9** ± 2.4	$-17.8(.0002)$	16.3 ± 3.6	$-\ 1.0(.895)$

Table 10: The averages and standard deviations of clarification questions that agents asked. The percent change (%Δ) and p-values are also shown. Changes of at least 10% percent are highlighted in blue, while values with $p < 0.05$ are **bolded**.

unproductive pairs like (*the lab*, *room_1*) and (*the lab*, *room_2*) from being used for training since in this case *the lab* was underspecified and did not accurately refer to either room.

4.2.3 Quantitative Performance

Tables 10 and 11 show agent performance in the untrained condition ($\mathcal{DA}_1$), the trained condition where *only* the perception modules were retrained ($\mathcal{DA}_4^*$), and the trained condition where *both* the parsing and perception modules were retrained ($\mathcal{DA}_4$). Shorter conversations with fewer clarification questions indicate the agent more quickly understood the task, while fewer enumeration fallback questions indicate that the agent less often used manual feedback versus natural language. We hypothesized that the number of clarification and enumeration fallback questions would decrease as more conversations became available for training the agent. We conducted a Welch's t-test to compare the number of $\mathcal{DA}_4^*$ (*Perception*) and $\mathcal{DA}_4$ (*Parsing+Perception*) agent questions against the number of $\mathcal{DA}_1$ agent questions.

Results. For the *navigation* task, we found that further training from conversation data did not lower the number of clarification or enumeration questions. This is similar to the result demonstrated in our past work for navigation tasks, and may be due to the small number of semantic roles (Thomason et al., 2015).

	Enumeration Questions ↓					
	Navigation		Delivery		Relocation	
Agent	#Qs	%Δ (p)	#Qs	%Δ (p)	#Qs	%Δ (p)
$\mathcal{DA}_1$	0.78 ± 0.41		1.26 ± 0.44		2.00 ± 1.00	
$\mathcal{DA}_4^*$	0.73 ± 0.45	$-6.4(.62)$	1.33 ± 0.52	$5.6(.464)$	1.95 ± 0.79	$-2.5(.89)$
$\mathcal{DA}_4$	0.82 ± 0.39	$5.1(.73)$	$\mathbf{0.91} \pm 0.50$	$-27.8(.001)$	1.80 ± 0.60	$-10(.60)$

Table 11: The averages and standard deviations of enumeration fallback questions agents asked. The percent change (%Δ) and p-values are also shown. Changes of at least 10% percent are highlighted in blue, while values with $p < 0.05$ are **bolded**.

For the *delivery* task, the real world object to be delivered and the recipient person needed to be communicated. Training the perception module alone ($\mathcal{DA}_4^*$) contributed most of the reduction in clarification questions, indicating that selecting the correct object was more difficult than identifying the recipient. The decreases in clarification questions for $\mathcal{DA}_4^*$ and $\mathcal{DA}_4$ were statistically significant. Adding parser training ($\mathcal{DA}_4$), the reduction in enumeration questions was also statistically significant. The $\mathcal{DA}_4$ agent could memorize additional names and titles for potential recipients.

For the *relocation* task, the real world object to be relocated and the source and goal locations needed to be communicated. Training the perception module alone ($\mathcal{DA}_4^*$) from previous conversations statistically significantly reduced the number of clarification questions but had little effect on enumeration questions. The $\mathcal{DA}_4^*$ agent could better understand the real world object to be relocated, giving an absolute gain of 2.4 questions, similar to the 2.3 absolute gain on the *delivery* task for identifying the target object. Enumeration questions, by contrast, were used heavily by the $\mathcal{DA}_4^*$ agent to identify the source and goal locations, averaging nearly 2 enumeration questions per dialog. Training the perception and parsing module together ($\mathcal{DA}_4$) reduced the number of enumeration questions by about 10%, helping to resolve some location referring expressions. This parser training also added noise to understanding the target object referring expressions, offsetting the benefits the $\mathcal{DA}_4^*$ agent made in clarification question reduction.

4.2.4 User Survey

Table 12 measures users' experience with the agents in terms of feeling understood or frustrated. Table 13 shows user responses to prompts about whether they would use the robot to accomplish tasks in the real world. We hypothesized that users would find the system more understanding, less frustrating, and more usable as more conversations became available for training the agent. We conducted a Welch's t-test to compare the $\mathcal{DA}_4^*$ (*Perception*) and $\mathcal{DA}_4$ (*Parsing+Perception*) agent ratings to the $\mathcal{DA}_1$ agent ratings. We note that while no survey question changes between agents are statistically significant at $p < 0.05$, the number of workers is relatively small, and some p-values are suggestive (e.g., $p = 0.13$ for the usability of the *relocation* task).

Results. Training perception ($\mathcal{DA}_4^*$) caused a small change in how understood users felt (5.3%), with a larger reduction in user frustration (6.7%). With the addition of parser training ($\mathcal{DA}_4$), users felt more understood (10.5%) and similarly less frustrated (4.4%),

	Survey (Likert 1-7)			
	Understood ↑		Frustrated ↓	
Agent	Response	%Δ (p)	Response	%Δ (p)
$\mathcal{DA}_1$	3.8 ± 1.8		4.5 ± 1.7	
$\mathcal{DA}_4^*$	4.0 ± 1.8	5.3(.63)	4.2 ± 1.7	−6.7(.28)
$\mathcal{DA}_4$	4.2 ± 1.7	10.5(.24)	4.3 ± 1.8	−4.4(.37)

Table 12: Survey prompt averages for users feeling understood by or frustrated by the agent. The percent change (%Δ) and p-values are also shown. Changes of at least 10% percent are highlighted in blue.

	Usability Survey (Likert 1-7) ↑					
	Navigation		Delivery		Relocation	
Agent	Response	%Δ (p)	Response	%Δ (p)	Response	%Δ (p)
$\mathcal{DA}_1$	2.8 ± 1.8		3.3 ± 1.8		2.9 ± 1.4	
$\mathcal{DA}_4^*$	3.4 ± 1.9	21.4(.28)	3.2 ± 1.7	−3.0(.77)	3.1 ± 2.0	6.9(.75)
$\mathcal{DA}_4$	3.1 ± 1.7	10.7(.55)	3.8 ± 1.7	15.2(.18)	3.8 ± 1.8	31.0(.13)

Table 13: Usability survey averages for each task across the agents. The percent change (%Δ) and p-values are also shown. Changes of at least 10% percent are highlighted in blue.

though none of these differences were statistically significant. The $\mathcal{DA}_4^*$ and $\mathcal{DA}_4$ agents averaged around score 4 (*Neutral*) for both questions, while the initial $\mathcal{DA}_1$ agent, at score 4.5 for frustration, was closer to *Slightly Agree*.

For **usability**, the $\mathcal{DA}_4$ agent that retrained both parsing and perception modules received ratings 10% or higher per task than the initial $\mathcal{DA}_1$ agent, while the $\mathcal{DA}_4^*$ agent did the same for the *navigation* task only. These results were not statistically significant. The $\mathcal{DA}_4$ agent was rated near score 4 (*Neutral*) for the *delivery* and *navigation* usability, improving over the $\mathcal{DA}_1$ average, which was near 3 (*Slightly Disagree*) for both tasks.

4.2.5 Learned Perceptual Concept Models

During training, the agent acquired new perceptual concept models (22 in total) and new synonym words for existing concepts. The learned concept models were noisy.[8] Nonetheless, these learned models quantitatively and qualitatively improved user experience with the agent. Table 14 shows the distribution of confidence for some of the learned perceptual concept models.

Eight models simply returned the majority class seen at training time regardless of the object in consideration. By contrast, 14 learned to classify input objects. Therefore, we examined classification confidence based on sensorimotor context classifiers. These 14 learned models were: *container*, *blue*, *can*, *coffee*, *cylinder*, *empty*, *long*, *metallic*, *red*, *styrofoam*, *tower*, *wipes*, *white*, and *yellow*. These concept models achieved an average $\kappa = .86$ with

8. For example, seven workers labeled a uniformly yellow mustard container as a negative example of *yellow*. Fortunately, nine correctly identified it as a positive example.

Learned Perceptual Concept Models' Distributions

Sensible Models

can:

0.27	0.21	0.19	0.15	0.12	0.06	0.00	0.00

long:

0.66	0.16	0.10	0.08		+ Train	− Train	...

Noisy Models

blue:

0.67	0.33		+ Train	− Train	− Train	− Train	...

tower:

0.78	0.22		+ Train	− Train	− Train	− Train	...

Table 14: Sample perceptual concept models from conversations with human users. The numbers below each object in the active test set represent the probability of the word applying to the object based on learned concept models. For models assigning many objects zero confidence (e.g., the models predict that the word does not apply), we give samples of the positive and negative object examples labeled during conversations.

the majority vote of human annotators per object when evaluated using leave-one-out cross validation on the training objects. The average agreement with human annotators for all 22 learned concept models (including those for which the majority class was always returned) was $\kappa = 0.91$.

4.2.6 Language Analysis

Table 15 summarizes the average length of commands and unique token counts. The *delivery* task elicited longer commands with more word types than the *navigation* task, and the *relocation* task elicited the longest commands and most word types. Clarification questions (e.g., open-ended and yes/no questions posed by the agent) elicited more token types than initial commands, suggesting they contain higher lexical diversity useful for retraining

	Navigation	Delivery	Relocation	All
Command Length (Tokens)	6.4 ± 4.0	6.6 ± 3.5	11.2 ± 8.7	8.2 ± 6.4
Unique Command Tokens	245	237	344	495
Unique Clarification Tokens	448	548	595	976
Unique Tokens (All)	508	590	682	1086
Test Time Unseen Tokens	178	170	237	351

Table 15: Token lengths and unique counts of commands and clarifications from human users at training and test time. We also show the unique, unseen tokens at test time.

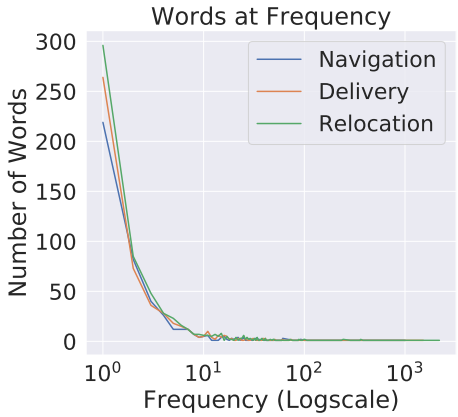


Figure 16: The word counts across all tasks from human user utterances binned by frequency. The x-axis shows the number of times a word token occurred in the data, while the y-axis shows the number of words at that frequency.

the parser. Finally, many tokens were unique and unseen at test time, with the proportion of unseen token types averaging 32.3% across all test time utterances. This fraction indicates the difficulty of generalizing to unseen tasks at test time. Such tasks introduce unseen vocabulary for the natural language understanding pipeline that must be handled via embedding nearest neighbors and clarification questions.

Figure 16 shows the distribution of word counts across tasks. The distribution is Zipfian, with the vast majority of word types occurring fewer than 10 times (far left-hand side). The most frequent words, occurring over 100 times, number only a few (right-hand side). Infrequent words can have impoverished representations. This is a well-known issue for natural language applications, which we ameliorate by leveraging pre-trained word embeddings to fill in knowledge gaps for unseen words.

We examined words used during training dialogs versus test dialogs. Figure 17 displays the top 50 words with the largest proportional difference (difference in relative frequency per condition) in frequency between training and test user utterances. We observe that test words include people (e.g., *nancy nagel*), room references (e.g., *conference*), and object descriptor words (e.g., *pringles*, *white*) that are infrequent in the training data. These proportional differences expose difficult test time words.

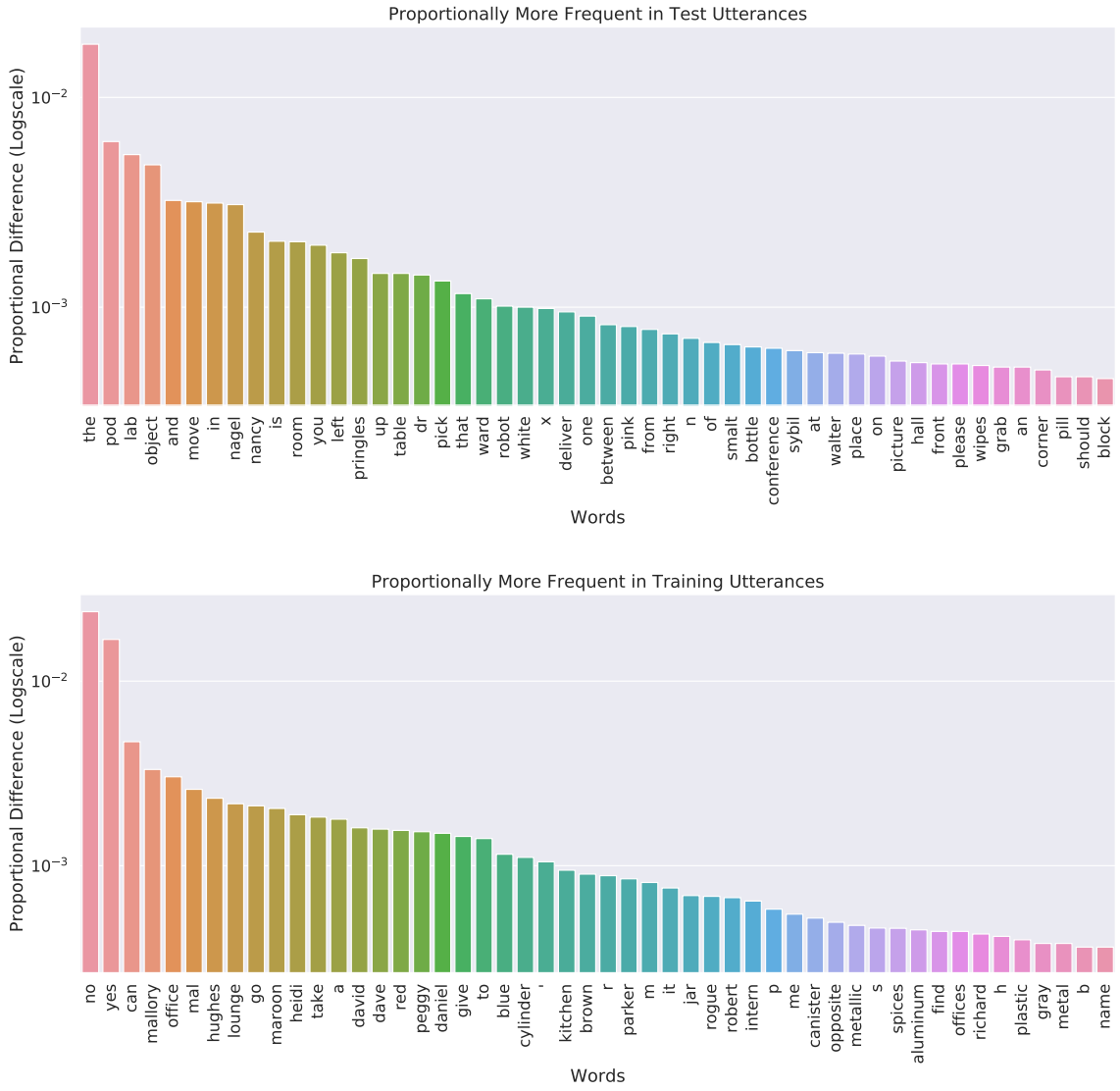


Figure 17: Words with the largest proportional difference in frequency between user utterances in training versus test conditions. Note that *yes* and *no* are more common in training utterances. During training, the agent refined its perceptual concept models with yes/no label questions.

4.2.7 Pipeline Time-Outs

Latency in conversation response time is key for smooth interaction. We imposed a 15-second restriction on the agent's parsing module and a 10-second restriction on the agent's grounding module. These restrictions prevented the parser from understanding some utterances, but they saved wasted effort on utterances without reachable parse trees. Grounding deep recursive structures (e.g., *the office next to the empty office next to the eastern lab*) was also less likely to succeed against the time constraint.

The parser timed out on 1711 (13%) and the grounder on 173 (1.3%) of the 13188 utterances. There was little variation in these rates between training and test conditions. Most failures were utterances that were verbose (e.g., *go to richard rogue's office and retrieve a can of peaches leave richard rogue's office and take it two offices down to peggy parker's office*). These modules processed information more slowly when running on-device for a deployed robot. Thus, for the robot demonstration, we removed processing time limits. We note that in practice, natural language understanding components must run and respond in a realistic and small time window, achievable by running on a remote server, for example.

4.3 Physical Robot Demonstration

We used a BWIBot (Khandelwal, Yang, Leonetti, Lifschitz, & Stone, 2014; Khandelwal et al., 2017) equipped with a Kinova MICO arm to manipulate objects; an ASUS Xtion Pro camera to view objects on tabletop surfaces; a Hokuyo lidar to perform obstacle avoidance; and a Blue Snowball microphone for recording speech (Figure 6). An Alienware computer executed onboard computation. The robot used a mobile Segway base reinforced with two additional 12V Li-Ion batteries to power the base, arm, computer, and sensors.

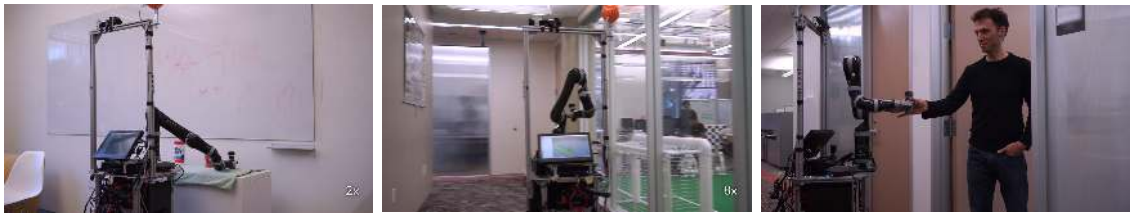
The BWIBot software stack provided automated task planning and autonomous navigation. For this demonstration, speech recognition was provided by the Google Speech API (Google, 2020) and speech synthesis was performed with the Festival Speech Synthesis System (Festival, 2020). Tabletop perception for both dialog interaction and execution of the resulting command was implemented with RANSAC (Fischler & Bolles, 1981) plane fitting and Euclidean clustering provided by the Point Cloud Library (Rusu & Cousins, 2011). The robot approached tables by selecting the largest horizontal planar surface in its view and moving to a fixed distance from the nearest point of the plane. Manipulation was realized using a heuristic grasping approach. Candidate grasps were generated along the bounding box of the perceived object, filtered for their reachability under the robot's kinematics, then ranked to prefer grasps near the object's centroid. The robot executed handovers by presenting the item and waiting for its joint efforts to exceed a threshold. The symbolic planning domain assumed a well-defined goal location (A.1).

In the demonstration (Figure 18), the dialog began near a set of *active training set* objects. A human user instructed the robot to *Move a rattling container from the lounge by the conference room to Bob's office*. The dialog agent was initialized using predicates learned from Amazon Mechanical Turk interactions, but it did not know the word *rattling*. The agent checked word embedding to see if it could relate the word to a known predicate. After the human rejected the agent's synonymy queried, it queries the user for positive and negative examples of rattling items from nearby objects. After this exchange, the robot confirmed that the user intended to issue an object relocation command and that it grounded the goal location correctly. The agent trained a new perceptual classifier for the *rattling* predicate and correctly identified the requested item from among several candidates in the source room. These candidates were already explored (e.g., had feature representations) because we assumed a closed-world of objects (A.5).

The grounded command action, object, source room, and goal room were passed to task level control (A.1). The task was decomposed into three parts: navigation to the source room, retrieval of the object, and navigation to the goal room for a handover. In this

R: What should I do?
H: Move a rattling container from the lounge by the conference room to Bob's office.
R: Does rattling refer to a property of an object?
H: Yes.
R: Does rattling mean the same thing as empty?
H: No.
R: I am still learning the meanings of some words.
R: Show me an object you could use the word rattling when describing, or say "none of them."
H: Watch.
R: I am getting the objects on the table into focus... Okay, I see them.
H: This one. *The human picks up a small, full pill jar*
R: I see.
R: Show me an object you could not use the word rattling when describing, or say "all of them."
H: Watch. This one. *The human picks up a red water bottle*
R: I see.
R: Show me an object you could use the word rattling when describing, or say "none of them."
H: Watch. This one. *The human picks up a container half-full of beans*
R: I see.
R: Thanks. Now back to business.
R: You want me to move an item from 3.514 to somewhere?
H: Yes.
R: I should move something from 3.514 to 3.510?
H: Yes.

(a) The human says to move a *rattling container* from one room to another. (b) The robot asks questions about local items to learn the new *rattling* property. (c) The robot autonomously navigates to the specified location to search for the object.



(d) The robot decides which item is a *rattling container* and grasps it. (e) The robot autonomously navigates to the specified destination room. (f) The robot hands over the item at the specified destination.

Figure 18: A demonstration of the learning agent embodied on the physical BWIBot platform. The agent was pre-trained from a Mechanical Turk experiment presented in our recent work (Thomason et al., 2019), similar to that presented in this article. The agent learns a new word, *rattling*, and uses this new concept model to correctly identify and move the target item. Video: `https://youtu.be/PbOfteZ_CJc`

demonstration, navigation to the source room was realized with three symbolic actions: approaching the door, going through the door, and going to a position in the lounge. On arrival at the source location, the robot perceived its surroundings and executed a table approach. The robot identified the objects on the table, ran trained perceptual concept models for *container* and *rattling* to select the most probable referent object, and executed an object grasp. Once the object was in the gripper, the robot navigated to face the goal room door. At the goal door, the robot handed over the object to Bob.

5. Conclusion

This article presents a dialog agent (C.1) that improves both semantic parsing (C.2) and grounded perception (C.3). This agent uses clarification questions to refine its understanding of user intent, then aligns answers to those questions with underlying denotations to improve its semantic parser. The agent asks questions about local objects and their physical properties selected through an opportunistic active learning strategy.

5.1 Summary

Via a large-scale Mechanical Turk experiment, we demonstrate that the agent needs to ask users fewer clarification and enumeration questions after learning from conversations with previous users. In particular, we find that the $\mathcal{DA}_4^*$ agent with a trained perception module and $\mathcal{DA}_4$ agent with both trained perception and parsing modules achieved reductions in either the number of clarification or enumeration questions in at least one task when compared to the $\mathcal{DA}_1$ agent. Overall, the $\mathcal{DA}_4$ agent conducted shorter conversations with fewer tedious enumeration questions on the *delivery* task, and it additionally reduced the number of such enumeration questions by 10% on the *relocation task*. Further, we find that users rated the overall system more usable for real-world tasks after conversation-based learning. In particular, the $\mathcal{DA}_4$ agent received usability ratings that were 10% or more higher than the $\mathcal{DA}_1$ initial agent across all three tasks. Finally, we embodied this learning agent, initialized from Mechanical Turk conversations, in a physical robot platform to demonstrate its learning abilities for the non-visual word *rattling*, which was not yet learned during training, to execute a novel test command.

5.2 Future Work

The assumptions enumerated in Section 1 represent limitations that could be overcome in future work. We assume that actions the agent can perform can be broken down into semantic tuples (A.1), and that these actions are discrete (A.4). This kind of semantic role-based understanding, while well-studied, does not easily extend to continuous control (e.g., *go slowly*) or facilitate constraints (e.g., *go to bob's office but avoid the kitchen*). Future work might instead perform command understanding by inferring a set of post-conditions for a planner that include constraints and modifiers (Nyga et al., 2018).

Leveraging transfer learning between similarly deployed robots (for example, in different hospitals) could increase the human-robot language data available per robot when human-robot conversations take place on multiple robot platforms. We have used only English language data in this article. However, when sharing between robots, multiple languages

may need to be understood simultaneously, for example, by sharing perceptual representations grounded in one language to language tokens in another (Kery, 2018). Transferring embodied language representations between robots that are morphologically different, however, remains a challenge, e.g., our robot's classifier for the word *soft* would not be directly deployable on another robot with different degrees of freedom and different object exploration behaviors. One promising direction to explore in future work is the use of encoder-decoder networks for transferring sensorimotor knowledge related to language grounding from one robot to another (Tatiya, Hosseini, Hughes, & Sinapov, 2019). We hypothesize that learned manifold alignments (Tuia & Camps-Valls, 2016) between object feature representations from different platforms may also allow transfer of object experiences between distinct robots. Additionally, exhaustive exploration may be ameliorated by applying only relevant behaviors when exploring new objects, for example, using only the *look* behavior for the expression *the green cube* (Thomason, Sinapov, Mooney, & Stone, 2018).

We assume that users are cooperative and truthful (A.3) when answering questions. Our retraining procedure approximates this assumption (e.g., by discarding data from users who confirm completely incorrect tasks). However, noisy responses still muddle training data. For instance, users appear to pay less attention when selecting example objects, clicking *Shake Head*, implicitly labeling all shown objects as negative examples, even when positive ones exist in the set. It may help to infer user truthfulness during dialog (Vinanzi, Patacchiola, Chella, & Cangelosi, 2019) as an auxiliary goal, or to use user-specific dialog policies (Doering et al., 2019) to develop rapport (Marge & Rudnicky, 2019).

We assume a closed world (A.5), but in general an embodied agent will encounter new people, objects, and parts of an environment. There are active lines of research regarding environment exploration (Wang et al., 2018), object discovery (Tucker, Aksaray, Paul, Stein, & Roy, 2017), and identifying missing referents via dialog (Amiri, Bajracharya, Goktolga, Thomason, & Zhang, 2019). These strategies are compatible with our current dialog framework, which grounds to and asks enumeration questions about all known, relevant referents via a visual depiction achievable by taking photos of new referents. The closed world assumption allows us to calculate the distribution over possible object referents when considering perceptual concepts. However, as the number of objects grows, the scalability of this distributional strategy may be untenable. Hierarchical grounding using an external resource like WordNet (Fellbaum, 1998) to first identify whether an object is a dish (e.g., *mug*) or a food item (e.g., *spam*) before calculating the perceptual match of the reference words (e.g., for the word *mug*, consider only objects that can be more broadly classified as *dish*es) may help this approach scale. The closed world assumption also lets us split our agent's grounding into static knowledge versus perceptual knowledge. Relaxing this assumption may instead split knowledge into static (known) and dynamic (changing) knowledge, facilitating object instance changes over time, such as foods becoming sliced or cooked (Bullard, Schroecker, & Chernova, 2019).

We assume perceptual concepts are independent, categorical, and unique (A.6). Many natural language constructions are not grounded independently. For example, consider the referents of *fake gun*; the set of things describable as *fake* is excluded from things describable as *gun* (since they are, by nature of being fake, not guns). Additionally, many words are graded, not categorical, such as *heavy*. For example, a *heavy mug* is much lighter than a *heavy table*. Adjectives and nouns do not uniquely map to individual concepts. For example,

a *light mug* may be one that is light in weight or light in color since *light* is polysemous. In all of these cases, both linguistic and environmental context matter, and future work should incorporate multiple modalities of context (Thomason & Mooney, 2017).

Another promising area for future work is to further leverage learned, neural feature representations. We currently use the penultimate layer of the ResNet-512 network (He et al., 2016a) as a sensorimotor context space for *look*ing at objects. We could similarly use auto-encoders over object representations (Burchfiel & Konidaris, 2017) to provide a reduced feature vector representing a *learned*, rather than hand-crafted, feature space for every sensorimotor context. For example, features learned through an encoding for the matrix of haptic motor feedback during the lifting behavior through time (Tatiya & Sinapov, 2019) may prove richer than binning.

We hope that our agent and learning strategies for an end-to-end dialog system with perceptual connections to the real world inspire further research on grounded human-robot dialog for command understanding. Weakening or removing the assumptions made in this article are potential next steps towards embodied, conversational agents in environments shared with humans.

Acknowledgements

We would like to thank our anonymous reviewers for their detailed comments and suggestions, all of which strengthened this article. This work is supported by a National Science Foundation Graduate Research Fellowship to the first author, an NSF EAGER grant (IIS-1548567), and an NSF NRI grant (IIS-1637736). This work took place in the Learning Agents Research Group (LARG) at the Artificial Intelligence Laboratory, the University of Texas at Austin. LARG research is supported in part by grants from the National Science Foundation (CNS-1305287, IIS-1637736, IIS-1651089, IIS-1724157), the Texas Department of Transportation, Intel, Raytheon, and Lockheed Martin. Peter Stone serves on the Board of Directors of Cogitai, Inc. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research. Portions of this research were conducted under protocols approved by the University of Texas at Austin Institutional Review Board, protocol 2013-06-0044 with PI Peter Stone and protocol 2017-01-010 with PI Justin Hart.

References

A. Lazaridou, E. B., & Baroni., M. (2014). Is this a wampimuk? Cross-modal mapping between distributional semantics and the visual world. In *Association for Computational Linguistics (ACL)*.

Aldoma, A., Marton, Z.-C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., Rusu, R. B., Gedikli, S., & Vincze, M. (2012). Point cloud library. *IEEE Robotics & Automation Magazine*, *1070*(9932/12).

Alomari, M., Duckworth, P., Bore, N., Hawasly, M., Hogg, D. C., & Cohn, A. G. (2017). Grounding of human environments and activities for autonomous robots. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Amiri, S., Bajracharya, S., Goktolga, C., Thomason, J., & Zhang, S. (2019). Augmenting knowledge through statistical, goal-oriented human-robot dialog. In *International Conference on Intelligent Robots and Systems (IROS)*.

Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., & van den Hengel, A. (2018). Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Artzi, Y., & Zettlemoyer, L. (2011). Bootstrapping semantic parsers from conversations. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Artzi, Y., & Zettlemoyer, L. (2013a). UW SPF: The University of Washington Semantic Parsing Framework. In *arXiv:1311.3011*.

Artzi, Y., & Zettlemoyer, L. (2013b). Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics (TACL)*, *1*.

Bastianelli, E., Croce, D., Vanzo, A., Basili, R., & Nardi, D. (2016). A discriminative approach to grounded spoken language understanding in interactive robotics. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Berant, J., Chou, A., Frostig, R., & Liang, P. (2013). Semantic parsing on Freebase from question-answer pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Bisk, Y., Shih, K. J., Choi, Y., & Marcu, D. (2018). Learning interpretable spatial operations in a rich 3d blocks world. In *AAAI Conference on Artificial Intelligence*.

Bisk, Y., Yuret, D., & Marcu, D. (2016). Natural language communication with robots. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Bullard, K., Schroecker, Y., & Chernova, S. (2019). Active learning within constrained environments through imitation of an expert questioner. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Burchfiel, B., & Konidaris, G. (2017). Generalized 3d object representations using bayesian eigenobjects. In *Robotics: Science and Systems (RSS)*.

Chai, J. Y., Gao, Q., She, L., Yang, S., Saba-Sadiya, S., & Xu, G. (2018). Language to action: Towards interactive task learning with physical agents. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Chu, V., McMahon, I., Riano, L., McDonald, C. G., He, Q., Perez-Tejada, J. M., Arrigo, M., Fitter, N., Nappo, J. C., Darrell, T., et al. (2013). Using robotic exploratory procedures to learn the meaning of haptic adjectives. In *International Conference on Robotics and Automation (ICRA)*.

Culotta, A., & McCallum, A. (2005). Reducing labeling effort for structured prediction tasks. In *AAAI Conference on Artificial Intelligence*.

Dindo, H., & Zambuto, D. (2010). A probabilistic approach to learning a visually grounded language model through human-robot interaction. In *Intelligent Robots and Systems (IROS)*.

Doering, M., Liu, P., Glas, D. F., Kanda, T., Kulić, D., & Ishiguro, H. (2019). Curiosity did not kill the robot: A curiosity-based learning system for a shopkeeper robot. *Transactions on Human-Robot Interaction (THRI)*, *8*(15).

Dong, L., & Lapata, M. (2016). Language to logical form with neural attention. In *Association for Computational Linguistics (ACL)*.

Fahnestock, E., Patki, S., & Howard, T. M. (2019). Language-guided adaptive perception with hierarchical symbolic representations for mobile manipulators. In *Artificial Intelligence Fall Symposium Series (AAAI-FSS)*.

Fellbaum, C. D. (1998). *WordNet: An Electronic Lexical Database*. MITP, Cambridge, MA.

Festival (2020). The festival speech synthesis system. [Online]. Available from: `http://www.cstr.ed.ac.uk/projects/festival/`. [Accessed 2018].

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395.

FitzGerald, N., Artzi, Y., & Zettlemoyer, L. (2013). Learning distributions over logical forms for referring expression generation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Forbes, M., & Choi, Y. (2017). Verb physics: Relative physical knowledge of actions and objects. In *Association for Computational Linguistics (ACL)*.

Gao, Y., Hendricks, L. A., Kuchenbecker, K. J., & Darrell, T. (2016). Deep learning for tactile understanding from visual and haptic data. In *International Conference on Robotics and Automation (ICRA)*.

Google (2020). Cloud speech-to-text recognition. [Online]. Available from: `https://cloud.google.com/speech/`. [Accessed 2018].

Harnad, S. (1990). The symbol grounding problem. *Physica D*, *42*, 335–346.

He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*.

He, L., Michael, J., Lewis, M., & Zettlemoyer, L. (2016b). Human-in-the-loop parsing. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Hu, R., Xu, H., Rohrbach, M., Feng, J., Saenko, K., & Darrell, T. (2016). Natural language object retrieval. In *Computer Vision and Pattern Recognition (CVPR)*.

Iyer, S., Konstas, I., Cheung, A., Krishnamurthy, J., & Zettlemoyer, L. (2017). Learning a neural semantic parser from user feedback. In *Association for Computational Linguistics (ACL)*.

Jia, R., & Liang, P. (2016). Data recombination for neural semantic parsing. In *Association for Computational Linguistics (ACL)*.

Kery, C. (2018). Esta es una naranja atractiva: Adventures in adapting an english language grounding system to non-english data. Master's thesis, University of Maryland, Baltimore County.

Khandelwal, P., Yang, F., Leonetti, M., Lifschitz, V., & Stone, P. (2014). Planning in Action Language $\mathcal{BC}$ while Learning Action Costs for Mobile Robots. In *International Conference on Automated Planning and Scheduling (ICAPS)*.

Khandelwal, P., Zhang, S., Sinapov, J., Leonetti, M., Thomason, J., Yang, F., Gori, I., Svetlik, M., Khante, P., Lifschitz, V., Aggarwal, J. K., Mooney, R., & Stone, P. (2017). Bwibots: A platform for bridging the gap between ai and human–robot interaction research. *The International Journal of Robotics Research (IJRR)*, *36*.

Kiela, D., & Clark, S. (2015). Multi- and cross-modal semantics beyond vision: Grounding in auditory perception. In *Emperical Methods in Natural Language Processing (EMNLP)*.

Kočiskỳ, T., Melis, G., Grefenstette, E., Dyer, C., Ling, W., Blunsom, P., & Hermann, K. M. (2016). Semantic parsing with semi-supervised sequential autoencoders. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Kollar, T., Tellex, S., Roy, D., & Roy, N. (2010). Toward understanding natural language directions. In *Human-Robot Interaction (HRI)*.

Konstas, I., Iyer, S., Yatskar, M., Choi, Y., & Zettlemoyer, L. (2017). Neural AMR: Sequence-to-Sequence Models for Parsing and Generation. In *Association for Computational Linguistics (ACL)*.

Krishnamurthy, J., & Kollar, T. (2013). Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Association for Computational Linguistics (TACL)*, *1*.

Kwiatkowski, T., Choi, E., Artzi, Y., & Zettlemoyer, L. (2013). Scaling semantic parsers with on-the-fly ontology matching. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., & Steedman, M. (2010). Inducing Probabilistic CCG Grammars from Logical Form with Higher-Order Unification. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Lee, K., Lewis, M., & Zettlemoyer, L. (2016). Global Neural CCG Parsing with Optimality Guarantees. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Lewis, M., Lee, K., & Zettlemoyer, L. (2016). LSTM CCG Parsing. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Liang, P., Jordan, M. I., & Klein, D. (2011). Learning dependency-based compositional semantics. In *Association for Computational Linguistics (ACL)*.

Liang, P., & Potts, C. (2015). Bringing machine learning and compositional semantics together. *Annual Review of Linguistics*, *1*(1), 355–376.

Lynott, D., & Connell, L. (2009). Modality exclusivity norms for 423 object properties. *Behavior Research Methods*, *41*(2), 558–564.

Mao, J., Wei, X., Yang, Y., Wang, J., Huang, Z., & Yuille, A. L. (2015). Learning like a child: Fast novel visual concept learning from sentence descriptions of images. In *International Conference on Computer Vision (ICCV)*.

Marge, M., & Rudnicky, A. I. (2019). Miscommunication detection and recovery in situated human–robot dialogue. *Transactions on Interactive Intelligent Systems (TiiS)*, *9*(3).

Matuszek, C., Bo, L., Zettlemoyer, L., & Fox, D. (2014). Learning from unscripted deictic gesture and language for human-robot interactions. In *AAAI Conference on Artificial Intelligence*, Quebéc City, Quebéc, Canada.

Matuszek, C., FitzGerald, N., Zettlemoyer, L., Bo, L., & Fox, D. (2012a). A joint model of language and perception for grounded attribute learning. In *International Conference on Machine Learning (ICML)*.

Matuszek, C., Herbst, E., Zettlemoyer, L., & Fox, D. (2012b). Learning to parse natural language commands to a robot control system. In *International Symposium on Experimental Robotics (ISER)*.

Matuszek, C., Herbst, E., Zettlemoyer, L., & Fox, D. (2013). Learning to parse natural language commands to a robot control system. In *Experimental Robotics*, pp. 403–415. Springer International Publishing.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems (NeurIPS)*.

Mininger, A., & Laird, J. E. (2018). Interactively learning a blend of goal-based and procedural tasks. In *AAAI Conference on Artificial Intelligence*.

Misra, D., & Artzi, Y. (2016). Neural shift-reduce CCG semantic parsing. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Mohan, S., Mininger, A. H., Kirk, J. R., & Laird, J. E. (2012). Acquiring grounded representations of words with situated interactive instruction. In *Advances in Cognitive Systems*.

Mohan, S., Mininger, A. H., & Laird, J. E. (2013). Towards an indexical model of situated language comprehension for real-world cognitive agents. In *Conference on Advances in Cognitive Systems*.

Nyga, D., Roy, S., Paul, R., Park, D., Pomarlan, M., Beetz, M., & Roy, N. (2018). Grounding robot plans from natural language instructions with incomplete world knowledge. In *Conference on Robot Learning (CoRL)*.

Padmakumar, A., Stone, P., & Mooney, R. J. (2018). Learning a policy for opportunistic active learning. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Parde, N., Hair, A., Papakostas, M., Tsiakas, K., Dagioglou, M., Karkaletsis, V., & Nielsen, R. D. (2015). Grounding the meaning of words through vision and interactive gameplay. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Patki, S., Daniele, A. F., Walter, M. R., & Howard, T. M. (2019). Inferring compact representations for efficient natural language understanding of robot instructions. In *International Conference on Robotics and Automation (ICRA)*.

Paul, R., Arkin, J., Roy, N., & Howard, T. M. (2016). Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators. In *Robotics: Science and Systems*.

Paul, R., Barbu, A., Felshin, S., Katz, B., & Roy, N. (2017). Temporal grounding graphs for language understanding with accrued visual-linguistic context. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Paxton, C., Bisk, Y., Thomason, J., Byravan, A., & Fox, D. (2019). Prospection: Interpretable plans from language by predicting the future. In *International Conference on Robotics and Automation (ICRA)*.

Perera, I., & Allen, J. F. (2013). Sall-e: Situated agent for language learning. In *AAAI Conference on Artificial Intelligence*.

Pillai, N., & Matuszek, C. (2018). Unsupervised selection of negative examples for grounded language learning. In *AAAI Conference on Artificial Intelligence*.

Pizzuto, G., Hospedales, T., Capirci, O., & Cangelosi, A. (2019). Modelling the single word to multi-word transition using matrix completion. In *International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*.

Roy, D., & Pentland, A. (2002). Learning words from sights and sounds: a computational model. *COGSCI*, *26*(1), 113–146.

Rusu, R. B., Blodow, N., & Beetz, M. (2009). Fast point feature histograms (FPFH) for 3D registration. In *International Conference on Robotics and Automation (ICRA)*.

Rusu, R. B., & Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). In *International Conference on Robotics and Automation (ICRA)*.

Shah, P., Fiser, M., Faust, A., Kew, J. C., & Hakkani-Tur, D. (2018). Follownet: Robot navigation by following natural language directions with deep reinforcement learning. In *Conference on Robotics and Automation (ICRA) Third Workshop in Machine Learning in the Planning and Control of Robot Motion*.

Sinapov, J., Khante, P., Svetlik, M., & Stone, P. (2016). Learning to order objects using haptic and proprioceptive exploratory behaviors. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Sinapov, J., Schenck, C., Staley, K., Sukhoy, V., & Stoytchev, A. (2014a). Grounding semantic categories in behavioral interactions: Experiments with 100 objects. *Robotics and Autonomous Systems*, *62*(5), 632–645.

Sinapov, J., Schenck, C., & Stoytchev, A. (2014b). Learning relational object categories using behavioral exploration and multimodal perception. In *International Conference on Robotics and Automation (ICRA)*.

Skoviera, R., Stepanova, K., Tesar, M., Sejnova, G., Sedlar, J., Vavrecka, M., Babuska, R., & Sivic, J. (2018). Teaching robots to imitate a human with no on-teacher sensors. what are the key challenges?. In *International Conference on Intelligent Robots and Systems (IROS) Workshop on Towards Intelligent Social Robots: From Naive Robots to Robot Sapiens*.

Steedman, M., & Baldridge, J. (2011). Combinatory categorial grammar. In Borsley, R., & Borjars, K. (Eds.), *Non-Transformational Syntax: Formal and Explicit Models of Grammar*. Wiley-Blackwell.

Sun, Y., Bo, L., & Fox, D. (2013). Attribute based object identification. In *International Conference on Robotics and Automation (ICRA)*.

Tatiya, G., Hosseini, R., Hughes, M. C., & Sinapov, J. (2019). Sensorimotor cross-behavior knowledge transfer for grounded category recognition. In *International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*.

Tatiya, G., & Sinapov, J. (2019). Deep multi-sensory object category recognition using interactive behavioral exploration. In *International Conference on Robotics and Automation (ICRA)*.

Tellex, S., Knepper, R., Li, A., Rus, D., & Roy, N. (2014). Asking for help using inverse semantics. In *Robotics: Science and Systems (RSS)*.

Thomason, J., & Mooney, R. J. (2017). Multi-modal word synset induction. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Thomason, J., Padmakumar, A., Sinapov, J., Hart, J., Stone, P., & Mooney, R. J. (2017). Opportunistic active learning for grounding natural language descriptions. In *Conference on Robot Learning (CoRL)*.

Thomason, J., Padmakumar, A., Sinapov, J., Walker, N., Jiang, Y., Yedidsion, H., Hart, J., Stone, P., & Mooney, R. J. (2019). Improving grounded natural language understanding through human-robot dialog. In *International Conference on Robotics and Automation (ICRA)*.

Thomason, J., Sinapov, J., Mooney, R., & Stone, P. (2018). Guiding exploratory behaviors for multi-modal grounding of linguistic descriptions. In *AAAI Conference on Artificial Intelligence*.

Thomason, J., Sinapov, J., Svetlik, M., Stone, P., & Mooney, R. (2016). Learning multimodal grounded linguistic semantics by playing "I Spy". In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Thomason, J., Zhang, S., Mooney, R., & Stone, P. (2015). Learning to interpret natural language commands through human-robot dialog. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Tucker, M., Aksaray, D., Paul, R., Stein, G. J., & Roy, N. (2017). Learning unknown groundings for natural language interaction with mobile robots. In *International Symposium of Robotics Research (ISRR)*.

Tuia, D., & Camps-Valls, G. (2016). Kernel manifold alignment for domain adaptation. *PLoS ONE*, *11*.

Vanzo, A., Croce, D., Bastianelli, E., Basili, R., & Nardi, D. (2019). Grounded language interpretation of robotic commands through structured learning. *Artificial Intelligence*, *278*.

Vanzo, A., Part, J. L., Yu, Y., Nardi, D., & Lemon, O. (2018). Incrementally learning semantic attributes through dialogue interaction. In *Autonomous Agents and MultiAgent Systems (AAMAS)*.

Vinanzi, S., Patacchiola, M., Chella, A., & Cangelosi, A. (2019). Would a robot trust you? Developmental robotics model of trust and theory of mind. *Philosophical Transactions of the Royal Society B*, *374*(1771).

Vogel, A., Raghunathan, K., & Jurafsky, D. (2010). Eye spy: Improving vision through dialog. In *Association for the Advancement of Artificial Intelligence*, pp. 175–176.

Walter, M., Hemachandra, S., Homberg, B., Tellex, S., & Teller, S. (2013). Learning semantic maps from natural language descriptions. In *Robotics: Science and Systems (RSS)*.

Wang, S. I., Liang, P., & Manning, C. D. (2017). Learning language games through interaction. In *Association for Computational Linguistics (ACL)*.

Wang, X., Xiong, W., Wang, H., & Yang Wang, W. (2018). Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *The European Conference on Computer Vision (ECCV)*.

Whitney, D., Eldon, M., Oberlin, J., & Tellex, S. (2016). Interpreting Multimodal Referring Expressions in Real Time. In *International Conference on Robotics and Automation (ICRA)*.

Whitney, D., Rosen, E., MacGlashan, J., Wong, L. L. S., & Tellex, S. (2017). Reducing errors in object-fetching interactions through social feedback. In *International Conference on Robotics and Automation (ICRA)*.

Williams, E. C., Gopalan, N., Rhee, M., & Tellex, S. (2018). Learning to parse natural language to grounded reward functions with weak supervision. In *International Conference on Robotics and Automation (ICRA)*.

Yang, J., Lu, J., Lee, S., Batra, D., & Parikh, D. (2018). Visual curiosity: Learning to ask questions to learn visual recognition. In *Conference on Robot Learning (CoRL)*.

Younger, D. (1967). Recognition and parsing of context-free languages in time n^3. *Information and Control*, *10*, 189–208.

Zitnick, L., & Parikh, D. (2013). Bringing semantics into focus using visual abstraction. In *Computer Vision and Pattern Recognition (CVPR)*.